AD-A079 623 ANALYSIS AND COMPUTER SYSTEMS INC BURLINGTON MA F/G 17/1
DEVELOPMENT OF NUMERICAL TECHNIQUES AND COMPUTER SYSTEMS FOR DE--ETC(U)
AUG 79 L J ELTERMAN F19628-76-C-0257
UNCLASSIFIED RADC-TR-79-181 NL

1 OF 4
ADA
079623

LEVEL II

(11)

ADA079623

RADC-TR-79-181
Final Technical Report
August 1979

DEVELOPMENT OF NUMERICAL TECHNIQUES AND COMPUTER SYSTEMS FOR DESIGN, ANALYSIS AND FABRICATION OF INTERDIGITAL TRANSDUCERS AS SURFACE ACOUSTIC WAVE DEVICES

Analysis & Computer Systems, Inc.

Lawrence J. Elterman

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441

80 1 21 028

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-79-181 has been reviewed and is approved for publication.

APPROVED:

ANDREW J. SLOBODNIK, JR.
Project Engineer

APPROVED:

WALTER ROTMAN, Chief
Antenna and RF Components Branch
Electromagnetic Sciences Divsion

APPROVED:

ALLAN C. SCHELL, Chief
Electromagnetic Sciences Division

FOR THE COMMANDER:

JOHN P. HUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (EEA), Hanscom AFB MA 01731. This will assist us in maintaining a current mailing list.

Do not return this copy. Retain or destroy.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

19 REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: 18 RADC-TR-79-181
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER: 9
4. TITLE (and Subtitle): 6 DEVELOPMENT OF NUMERICAL TECHNIQUES AND COMPUTER SYSTEMS FOR DESIGN, ANALYSIS AND FABRICATION OF INTERDIGITAL TRANSDUCERS AS SURFACE ACOUSTIC WAVE DEVICES
5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report, 24 May 76 - 31 March 78
6. PERFORMING ORG. REPORT NUMBER: N/A
7. AUTHOR(s): 10 Lawrence J. Elterman
8. CONTRACT OR GRANT NUMBER(s): 15 F19628-76-C-0257, F19628-78-C-0028
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Analysis & Computer Systems, Inc., 109 Terrace Hall Avenue, Burlington MA 01803
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 61102F, 63727F, 62702F, 2305J522, 46001444
11. CONTROLLING OFFICE NAME AND ADDRESS: Deputy for Electronic Technology (RADC/EEP), Hanscom AFB MA 01731
12. REPORT DATE: 11 Aug 1979
13. NUMBER OF PAGES: 303
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Same 12 299
15. SECURITY CLASS. (of this report): UNCLASSIFIED
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A
16. DISTRIBUTION STATEMENT (of this Report): Approved for public release~~, distribution unlimited.~~ 16 2305, 4600
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report): Same 17 J5, 14
18. SUPPLEMENTARY NOTES: RADC Project Engineer: Andrew J. Slobodnik (EEA)
19. KEY WORDS (Continue on reverse side if necessary and identify by block number):

Surface Acoustic Wave	Glass Master
Interdigital Transducer	Date Flow
Computer Aided Design	Surface Wave Diffraction
Saw Analysi	

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Computerized, mathematical and analytical techniques were developed, refined and applied to the description, analysis and synthesis of surface acoustic wave (SAW) transduction structures and devices. Efficient mapping techniques between the various description mathematical models and techniques in support of electron beam fabrication of SAW interdigital transducers were developed and refined. A number of computer programs were completed ranging in complexity and size from modifications to existing programs, to design and development of complex analytical techniques utilizing numerical control methodology.

DD FORM 1 JAN 73 1473

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

403410

EVALUATION

1. This report is the Final Technical Report on contracts F19628-76-C-0257 and F19628-78-C-0028. It covers the development, refinement and application of computerized, mathematical and analytical techniques to the description, analysis and synthesis of acoustic concepts, waves, transduction structures and devices. Two of the more significant achievements were the development and refinement of techniques for efficient interfacing with widely-used surface acoustic wave (SAW) interdigital transducer analysis programs and a complete computerized system for interdigital transducer master specification and generation using automated mask generation equipment available through various vendors.

2. SAW devices are finding increasing useage in military and other systems due to their advantages of small size and weight, low cost and reliability. Since nearly all SAW devices follow the standard procedures of design, analysis and fabrication this work has broad and significant applicability.

Andrew J. Slobodnik, Jr.

ANDREW J. SLOBODNIK, JR.
Project Engineer

PREFACE

The analysis, development and programming efforts described in this report were performed for:

Rome Air Development Center (RADC)
Deputy for Electronic Technology
Hanscom AFB, Ma 01731

The computer programs documented herein describe the analytical and developmental techniques implemented for each system in addition to detailed program writeups.

The computer programs discussed in this report have been delivered to the above organization and questions and/or requests should be directed to the referenced organization.

The author expresses his thanks to Andrew J. Slobodnik, Jr., Contract Monitor, RADC/EEA, whose technical guidance and support were invaluable in the preparation of this report.

In addition, the author would like to thank the following RADC and ACSI personnel who provided valuable assistance in the preparation of this report:

Roger Colvin, RADC/EE, Tom Szabo, RADC/EE, for information provided during the development and documentation of computer programs, Harry Dolan and John Fusco, ACSI, for support in writing parts of the report and editing services provided, and Marion Swidrak, ACSI, for the many hours spent in typing the report.

TABLE OF CONTENTS

Section		Page
	PREFACE	3
	LIST OF ILLUSTRATIONS	6
1.0	INTRODUCTION	9
2.0	DESCRIBING SAW DEVICES	10
3.0	PROGRAM CONVERT	14
4.0	PROGRAM CONVXY	84
5.0	SPECIFYING A SAW DEVICE MASTER (STEP AND REPEAT MACHINE)	90
6.0	SPECIFYING A SAW DEVICE MASTER (PATTERN GENERATOR))	127
7.0	PROGRAM TRPLOT	149
8.0	STANDARD SAW DEVICE ANALYSIS PROGRAMS	153
9.0	GENERAL INSERTION LOSS PROGRAMS	169
10.0	GENERAL TEKTRONIX INTERFACE PROGRAMS	183
11.0	SUPPLEMENTAL SYSTEM PROGRAMS	195
12.0	ELECTRON BEAM FABRICATION OF INTERDIGITAL TRANSDUCERS	216
	APPENDICES	233

LIST OF ILLUSTRATIONS

Figure		Page
2.1	Pad Description	12
2.2	Pad Length equal to Pad Width	13
3.1	Comb with Single Electrodes and No Dummy Elements	17
3.2	Comb with Double Electrodes and No Dummy Elements	18
3.3	Comb with Single Electrodes and Dummy Fingers	19
3.4	Comb with Double Electrodes and Dummy Fingers	20
3.5	A Single Electrode Comb	25
3.6	A Double Electrode Comb	26
3.7	Certain Combs Reversed in Multi Comb Device	27
3.8	A Fictitious Zero Gap	28
3.9	Example of Zero Gap	28
3.10	Transducer with Negative Ho[N] Values	29
3.11	Pathological Case of Impossible Ho[N] Value	30
3.12	Coded Transducer, Odd Number of Taps, Even Number of Fingers per Tap	32
3.13	Coded Transducer, Odd Number of Taps, Odd Number of Fingers per Tap	33
3.14	Coded Transducer with Dummy Fingers, Even Number of Taps, Even Number of Fingers per Tap	34
3.15	Coded Transducer with Double Electrodes, Even Number of Taps, Odd Number of Fingers per Tap	35
3.16	A Zero Tap	42
3.17	Transducer with Negative Ho[NTAP] Values	43
3.18	Pathological Case of Impossible Ho[NTAP] Value	44

LIST OF ILLUSTRATIONS (Cont.)

Figure		Page
3.19	Coded Transducer with CARR = True	51
3.20	Coded Transducer, LEVER = True, NF = 6, SINGLE = True, NCHIPS = 4	53
3.21	Transducer Using Non Zero Value of Variable "ADD"	55
3.22	Examples of Coded Transducers with FILL=False and FILL=True	58
3.23	A Coded Transducer with ENGAN Type Elements	59
3.24	Case 1. No Dummy Fingers	61
3.25	Case 2. Dummy Fingers	62
3.26	Comb with Single Electrodes and no Dummy Elements	66
3.27	Comb with Double Electrodes and No Dummy Fingers	69
3.28	Coded Transducer, NTAPS=4, NF=4	73
4.1	Single Electrode, No Dummy Comb	89
5.1	Records Written on a Tape Created by Program HUGHES	95
5.2	Double Electrode Comb with CONOUT = .False.	100
5.3	Fabrication of Apodized Transducer with Large Reticle	102
5.4	A Plot of a Glass Master Produced By the Simulation Program HUESCHK	103
5.5	A Plot of a Glass Master Produced by the HUESCHK Simulation Program	104
5.6	Simplified Diagram of a Two Comb Master where SIDEBY = .True.	106
5.7	Simplified Diagram of One Comb Master where PADONLY = .True.	107
5.8	Master with Short = .True.	108
5.9	Simplified Diagram of a Master Produced by Program HUGHESXY	117

LIST OF ILLUSTRATIONS (Cont.)

Figure		Page
5.10	HUESCHKTEK Picture	126
6.1	Records Written on a Tape Created by Program HUGHES	132
6.2	Positive and Negative Masters	138
6.3	Picture of a Negative Master from a Tape Created by ELECTROXY and Simulated by Program ELCHECK	145
7.1	Plot from Program TRPLOT	152
8.1	Illustration of X-Axis of Graph from Program GRAF	162
8.2	Illustration of Y-Axis of Graph from Program GRAF	163
8.3	GRAF Plot with Tones Option	164
9.1	Two Transducers in Parallel with Matching Inductors and Parasitic Elements	177
9.2	The Same Electrical Device as shown in Figure 9.1 Redrawn for Convenient Circuit Analysis	178
11.1	Ratio Example	196
12.1	Charge Distribution required for Fabricating an Interdigital Transducer	219
12.2	Orientation of Transducers on SAW Substrate When Fabricated by Electron Beam Lithography	224
12.3	Flow Diagram of AMR Program Logic	225

1.0 INTRODUCTION

Surface Acoustic Wave (SAW) devices are extremely useful in a wide variety of current applications, for example, Command, Control and Communications. In particular, SAW Bandpass Filters offer the advantages of small size and weight, low cost and high reproducibility, and are easily produced and used in the VHF and UHF frequency ranges where other types of filters are most limited.

RADC is currently pursuing a program on the design, analysis, fabrication and testing of SAW filters. In direct support of this program, Analysis & Computer Systems, Inc. (ACSI) has developed a computer system for the analysis and fabrication of SAW devices.

This system provides the interface between existing RADC analytical tools and recently developed analytical programs, and also plots of the devices, which can be used for verifying the correctness of the data. In addition, the system also produces magnetic and paper tapes utilized for the fabrication of these devices directly from the data used in the analysis. This eliminates many errors in fabrication which have occurred in the past.

The programs and procedures described in the following sections present the analysis and methodology implemented in support of these R&D efforts. The function of each computer program, its role in the computer system and the utilization of each in support of the data flow is discussed. The techniques to be followed to implement the various functions, the input required to utilize each program and the procedures to be followed by the user to efficiently utilize the system are also discussed.

The programs described, although performing independent functions, have been designed to form a basic computer system in order to provide system flexibility and fulfill the goals of the research performed. Detailed error checking and evaluation are performed at strategic processing steps in order to ensure system integrity and provide efficient operational capability.

2.0 DESCRIBING SAW DEVICES

2.1 Introduction

There are several existing input formats available to the design engineer for describing a SAW device, among which are the RADC Standard Format, the RADC Coded Format and the Raytheon Format. The two RADC formats specify the physical characteristics of a device, in a particularly simple form. They are also useful in a wide variety of design and analysis programs.

Other efforts by the Raytheon Research Division resulted in several programs (e.g. COMBS3A, MATCH3) for the analysis of SAW devices using an equivalent circuit model. These programs are high. useful to the SAW device designer, particularly in the analysis of s ond order effects (e.g. triple transit echo, reflection, etc.). This Raytheon model described the SAW device using a coordinate system.

The Raytheon format, although not particularly suited to the general needs of the SAW device engineer, is far more convenient from a programmers point of view. For this reason subsequent programs written by ACSI for device plotting and fabrication also use the Raytheon format. Thus the need exists for an interface between the RADC and Raytheon formats. This need is addressed by program CONVERT, which is capable of converting either of the RADC formats to the Raytheon format.

2.2 Device Specifications

A SAW device of the type to be treated here is comprised of from one to sixteen combs (usually two). To describe a SAW device, one must describe each comb in the device.

In describing a comb, a device engineer must specify the following parameters for each comb in the device.

1. Width of Pads (Assumption: Both Pads within a comb are the same width).

2. Vertical distances between the pads of each comb.

3. Width of fingers.

4. For each finger:

   A. Whether the finger starts from the bottom or top Pad.
   B. Length of finger.
   C. Relative distance of left edge of Finger from left edge of first finger (See Figures 2.1, 2.2).

5. The length of the Pads. Both Pads of a particular comb are always the same length. In the usual case the length of a Pad is calculated by taking the maximum of:

   A. Width of the Pad.
   B. Distance from left edge of the leftmost finger to the right edge of the rightmost finger.

*Throughout this report "Pad" and "Bus Bar" have the same meaning.

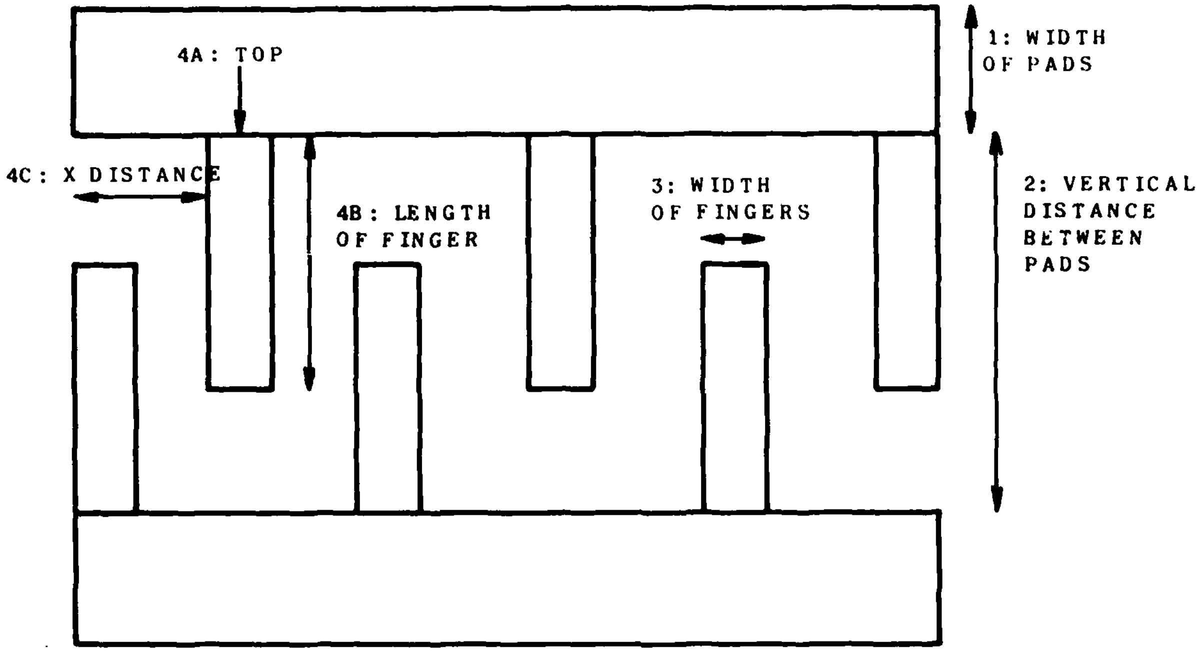


Figure 2.1 Pad Description

Length of Pad is the Distance from left edge of leftmost finger to right edge of rightmost finger. Also given are some examples of Device Specification as described in Section 2.2

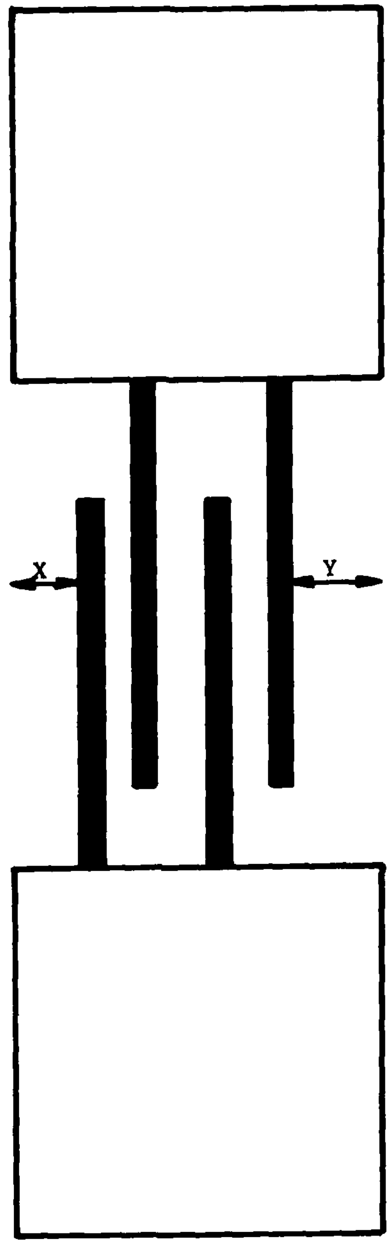


Figure 2.2: Pad Length equal to Pad Width. Fingers are centered horizontally on Pad. (i.e. X = Y)

3.0 PROGRAM CONVERT

3.1 Introduction

There are currently two main formats that RADC device engineers use to describe a SAW device; the RADC Standard Format, and the RADC Coded Format. Each of these two main formats have many variations and options. The purpose of these formats is to allow the device engineer to utilize a format that will closely parallel one's thought processes. The device engineer does not ordinarily think of a transducer in terms of a coordinate system. From a programmer's point of view, however, a coordinate system is very convenient. To develop programs to interpret and process the many RADC format variations, would be cumbersome and inefficient. Hence, program CONVERT was developed to allow the device engineer to utilize a familiar representative input format and still provide for efficient computer oriented implementation.

Program CONVERT accepts, as input, any of the many RADC format variations and produces a single standardized coordinate type output format, developed by Raytheon.

3.2 Input

Input to program CONVERT consists of the namelist $CONVERT and a set of cards for each comb in the device (maximum of five). The set of cards used to describe each comb may be any one of seven possible formats. They are:

A)	Ho[N] deck followed by ACT[M] deck	(RADC Standard Format)
B)	Ho[N] deck alone	(RADC Standard Format)
C)	ACT[M] deck alone	(RADC Standard Format)
D)	Ho[NTAP] deck followed by TCA[NTAP] deck	(RADC Coded Format)
E)	Ho[NTAP] deck alone	(RADC Coded Format)
F)	TCA[NTAP] deck alone	(RADC Coded Format)
G)	The Null Set (No Decks)	(RADC Coded Format)

Each comb need not be described by the same format; for example, in a three comb device the first comb might use Format A, the second comb Format D and the third comb Format G.

Details of these formats are described in the following sections.

3.3 Output

Output of program CONVERT consists of printed output, TAPE28,

TAPE29, and TAPE9 which contains a description of the device in the Raytheon format.

Details are presented in following sections.

3.4 The RADC Standard Format

The RADC standard format consists of a $CONVERT namelist followed by one or two decks for each comb in the device. The first deck is referred to as the Ho[N] deck (pronounced: H zero of N deck); the second deck is referred to as the ACT[M] deck.

The Ho[N] deck consists of a title card and a card for each gap between single electrodes, or in the case of a comb using double electrode pairs, it consists of one card for each gap between double electrode pairs. No cards are required for the gaps between fingers of double electrode pairs.

Each card following the title card consists of the number, N, (an integer) of the gap and Ho[N]*, (real) the amount of element** overlap at that gap. The inter-element gaps on the combs can be numbered from left to right, or from right to left, the restrictions being:

A) They must be numbered consecutively.

B) One of the gaps must be labeled gap zero.

C) The zero'th gap should contain the longest element overlap. Other gaps may have equal element overlap, but none should have more.

---

* The nomenclature used here is dual in nature. In general Ho[N] refers to the whole deck of cards described above. However, Ho[N] can also refer to the specific element overlaps of the gaps. Examples of both meanings follow:

A) I dropped an Ho[N] deck on the floor.
B) The Ho[N] value for the fifth gap equals .584 (i.e. Ho[5] = .584, or the fifth gap has an element overlap of .584).

**The nomenclature used in describing transducers is as follows. The word "element" is a general term that can refer to either a single fingered electrode or a double fingered electrode. "Double electrode" or "double electrode pair" refers to an element that has two fingers.

"Single electrode" refers to an element that has one finger.

The word "electrode" alone usually refers to a single fingered electrode, but can sometimes refer to a double electrode pair.

The words "finger" and "single electrode" can sometimes be used interchangeably.

From these conditions it is easy to see that some of the gaps must be numbered negatively. The gap numbered most negatively is called the minus 'NMIN'th gap, (pronounced: Minus N minth gap). The number NMIN is always positive. The gap labeled with the highest positive number is called the NMAX'th gap. The number NMAX is always positive (see Figures 3.1, 3.2, 3.3, 3.4).

When a comb has more than one gap with maximum overlap (often all the overlaps are equal), then it is up to the program user to decide which gap he wants to label zero. The overlap value on the card for the zero'th gap should always be 1. The other element overlap values are scaled accordingly. The actual overlap of the zero'th gap is defined by the variable "OVALAP" in the $CONVERT namelist. Thus, the actual length of the element overlap for any gap, N, is simply (OVALAP) (Ho[N]).

The format of the Ho[N] title card is 8A10; the format of the cards following the title card is (I5, E15.8).

3.4.1 Negative Ho[N] Values, Zero Ho[N] Values and Almost Zero Ho[N] Values

Ho[N] may range from values of negative one to values of positive one. Immediately the question comes to mind, "What is a negative element overlap?"

Physically there is no such thing as a negative element overlap, but mathematically the sign of the Ho[N] values change every time a zero overlap value occurs. That is, if Ho[N] = 0, then the signs of Ho[N-1] and Ho[N+1] will be opposite.

Now the question may be asked "What is a zero element overlap?"

A zero element overlap is not two elements on opposite bus bars that do not overlap (Figure 3.8), but rather two elements of the same length on the same bus bar (Figure 3.9). A change of sign cannot occur without a zero gap.

If a change of sign does occur in an Ho[N] deck without a zero Ho[N] value included, the program will automatically change one of the two Ho[N] values involved in the change of sign to zero.

The exact manner this is accomplished is specified by the following algorithm:

A) For N = -NMIN to NMAX

If Ho[N] $< 10^{-7}$ then Ho[N] = 0

B) For N = (-NMIN+1) to NMAX

If (Ho[N-1] < 0) and Ho[N] > 0) or

(Ho[N-1] > 0) and Ho[N] < 0)

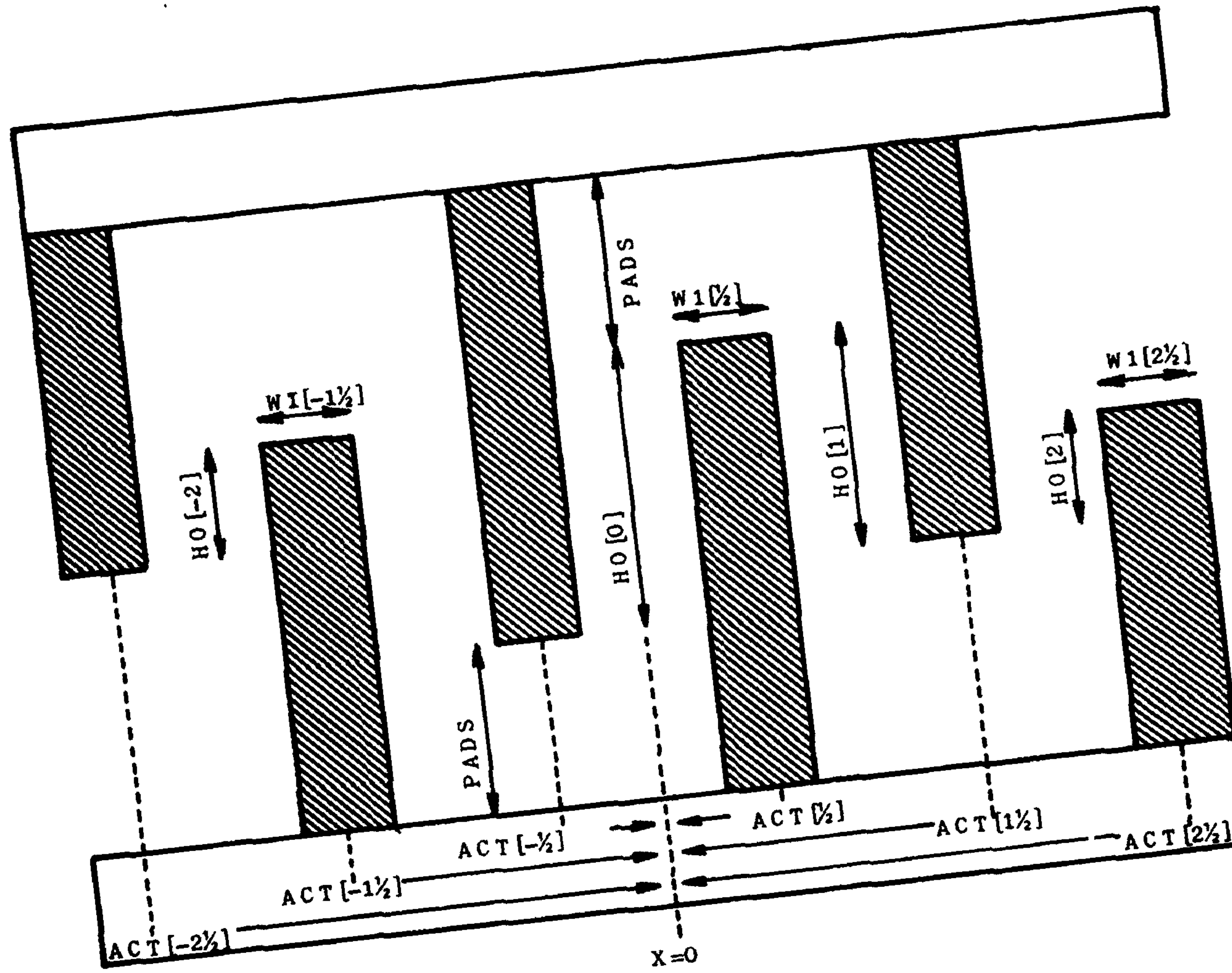


Figure 3.1 Comb with Single Electrodes and No Dummy Elements

Some examples of W1[M] shown. W2[M] and W3[M[are not defined in this case (single electrodes and no dummy elements). ACT[-2-1/2], ACT[-1-1/2], ACT[-1/2] are negative distances. NMIN=2, NMAX=2.

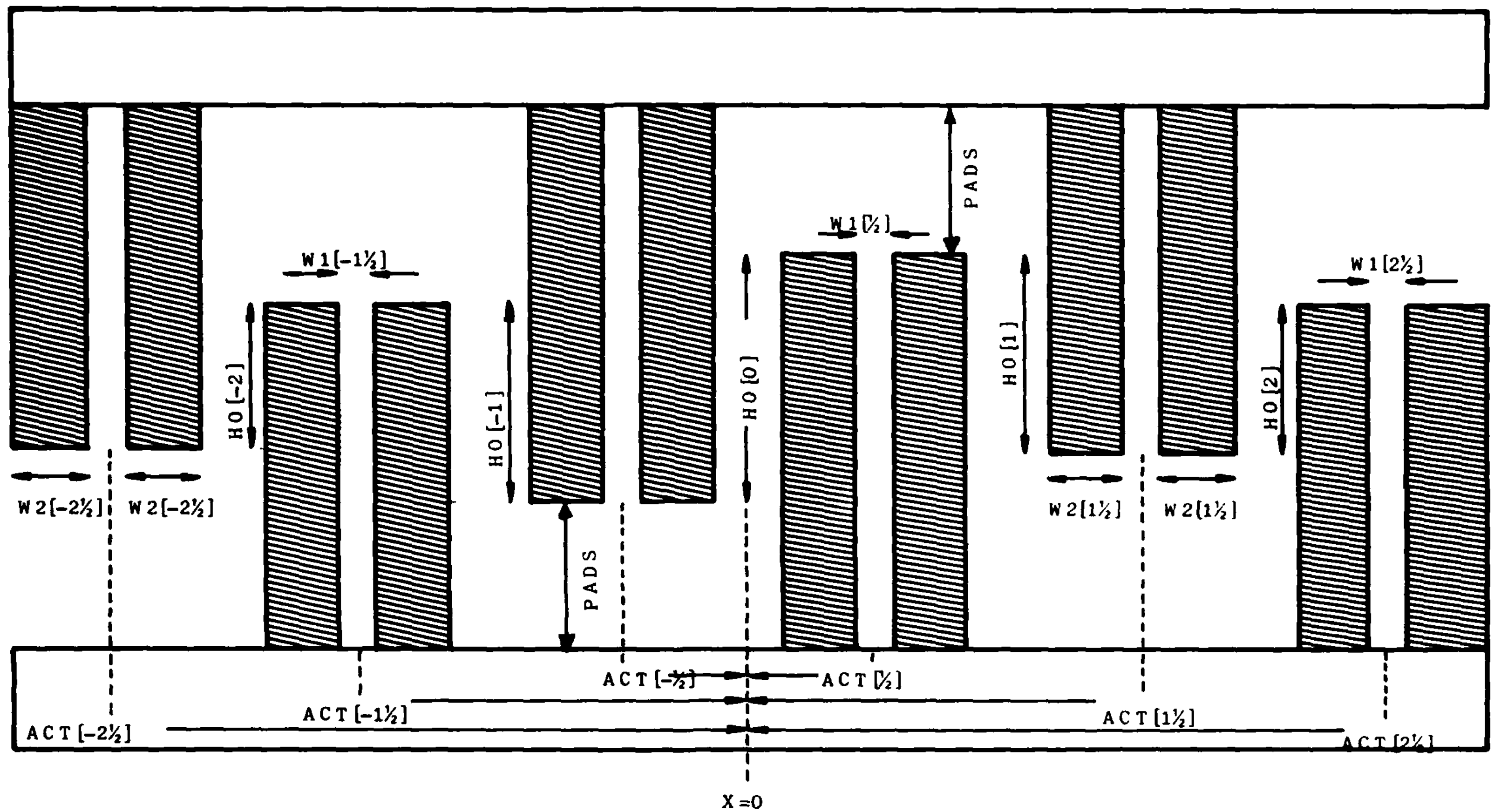


Figure 3.2 Comb With Double Electrodes and No Dummy Elements

Some example values of W1[M] and W2[M] are shown. W3[M] is not defined for this case (no dummy elements). ACT[-2-1/2], ACT[-1-1/2] and ACT[-1/2] are negative distances, NMIN=2, NMAX=2.

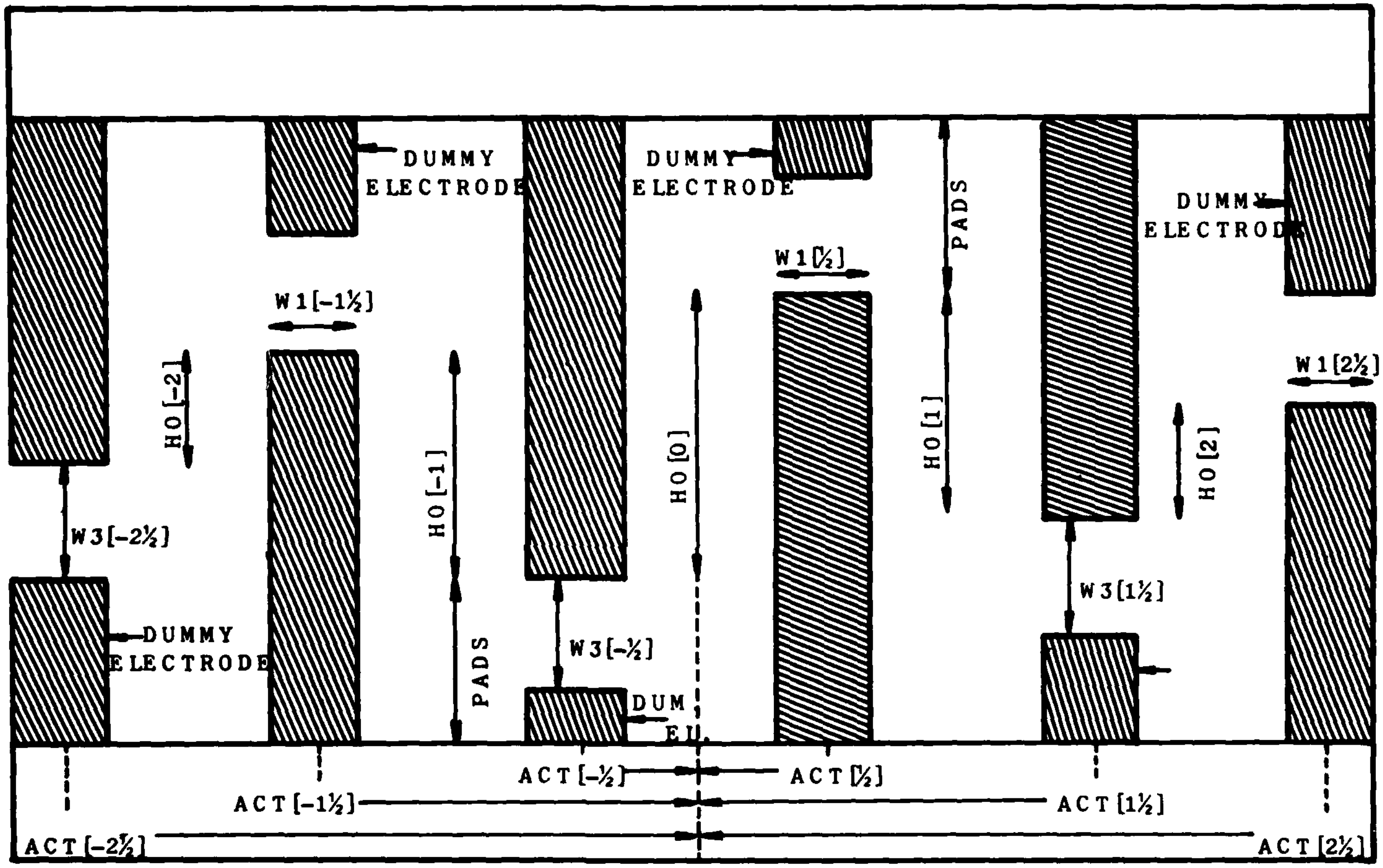


Figure 3.3 Comb with Single Electrodes and Dummy Fingers

Some examples of W1[M] and W3[M] are shown. W2[M] is not defined in this case (single electrodes). ACT[-2-1/2], ACT[-1-1/2], and ACT[-1/2] are negative distances, NMIN=2, NMAX=2. Dummy electrodes are electrically inactive.

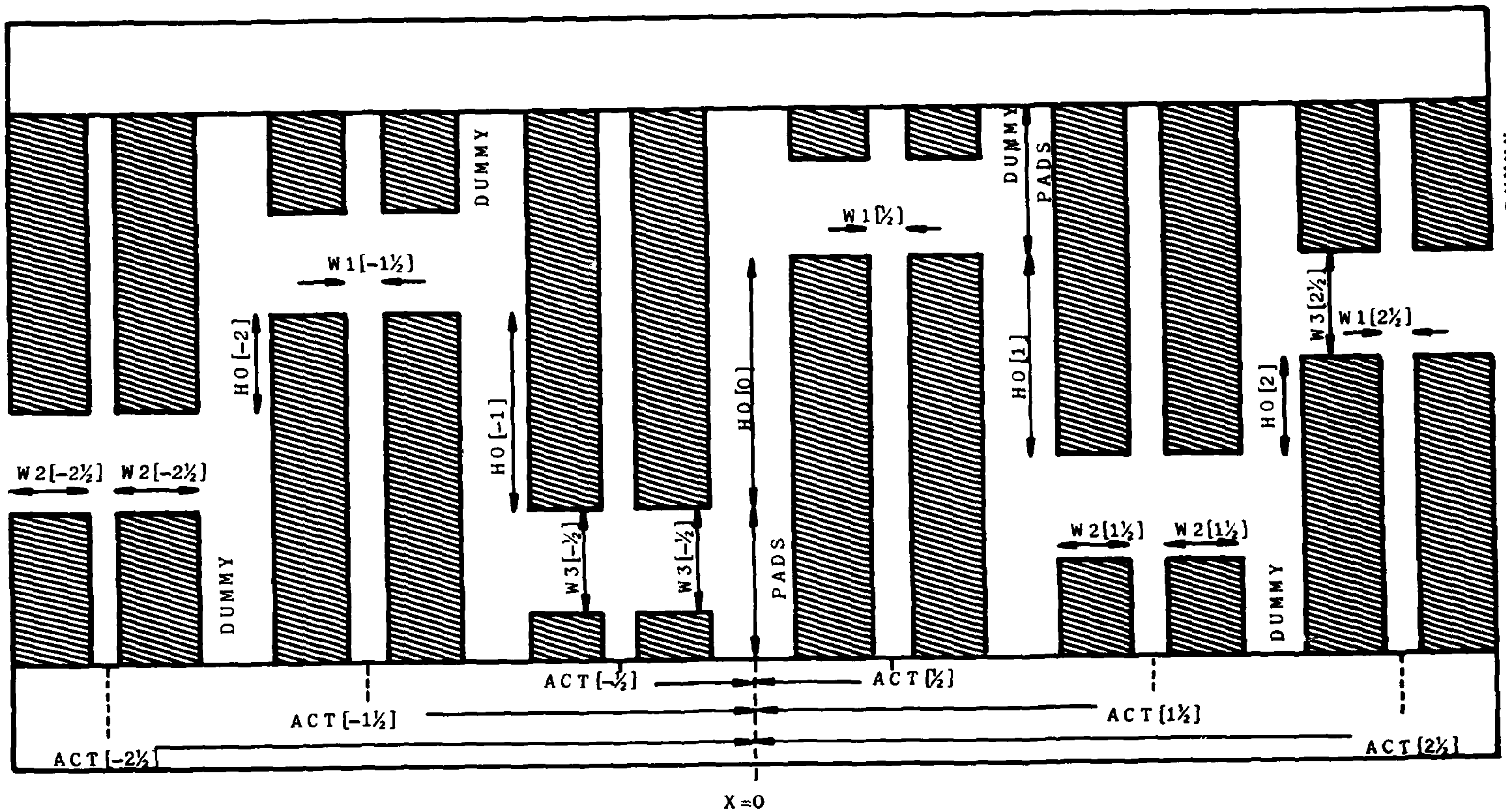


Figure 3.4 <u>Comb With Double Electrodes and Dummy Fingers</u>

Some example values of W1[M], W2[M], and W3[M] are shown. The double electrodes labeled "Dummy" are electrically inactive dummy elements. ACT[-2-1/2], ACT[-1-1/2], and ACT[-1/2] are negative distances, NMIN=2, NMAX=2.

Then

If ($|Ho[N-1]| \leq |Ho[N]|$) Ho[N-1] = 0.

If ($|Ho[N]| < |Ho[N-1]|$) Ho[N] = 0.

All Ho[N] decks are processed by this algorithm. Step A eliminates cases where Ho[N] is so close to zero as to possibly cause confusion. This algorithm is designed to handle the normal cases (examples C and D as follows), but not any pathological cases (examples E, F, G, H as follows).

	Ho[5]	Ho[6]	Ho[7]	Ho[8]	Ho[9]
C)	.7	.3	.1	-.2	-.4
D)	-.4	-.2	.1	.3	.7
E)	.7	.3	0	.3	.7
F)	-.7	-.3	0	-.3	-.7
G)	.7	.3	-.1	.3	.7
H)	-.7	-.3	.1	-.3	.7

The program will try to handle the pathological cases, but they will be handled incorrectly.

For a detailed example of a transducer with negative Ho[N] values see Figure 3.10.

3.4.2 The Pathological Case of Impossible Ho[N] Values

In the following example (Figure 3.11) it would be impossible to implement Ho[5]:

Ho[-2] = .3

Ho[-1] = .7

Ho[0] = 1.0

Ho[1] = .7

Ho[2] = .4

Ho[3] = .3

Ho[4] = .1

Ho[5] = .8

This case will probably never come up, but the user should be aware of its existence. In the event it does come up the program will continue to process incorrectly.

3.4.3 The ACT[M] Deck

The ACT[M] deck consists of a title card and a card for each electrode or double electrode pair. There is one more card in the ACT[M] deck than in the Ho[N] deck. This is because there is always one more element than the number of gaps between elements.

Each card following the title card consists of the number M, ACT[M]***, and optionally W_1[M], W_2[M] and W_3[M].

"M" is the number of the element. The first element on the negative side of the comb is numbered -NMIN-1/2. The following elements are numbered by adding increments of 1. Thus, the last element will be numbered NMAX + 1/2. Example: If we had a six element transducer the gaps might (but not necessarily) be numbered: (-2, -1, 0, 1, 2). The elements would then be numbered: (-2.5, -1.5, -.5, .5, 1.5, 2.5).

"ACT[M]" gives the distance from the center of the element to the middle of the zero'th gap. In the case of a double electrode, "Center" means, the center of the gap, that is between the two fingers of the double electrode. If the element is on the negative side of the comb (as defined by the number M) then this distance will be negative. If the element is on the positive side of the comb this distance will be positive.

"W1[M]": In the case of a single electrode W1[M]gives the width of the electrode. In the case of a double electrode W1[M] gives the width of the gap between the two fingers of the double electrode. If the W1[M] value is left off the card the program will use the W1 value from namelist $CONVERT.

"W2[M]": In the case of a single electrode W2[M] is not used, and will be ignored if given. In the case of a double electrode W2[M] is the width of one of the fingers of the double electrode. Both fingers of a double electrode are always assumed to be the same width. If the W2[M] value is left off the card, the program will use the W2 value from namelist $CONVERT.

***As with the case of Ho[N], ACT[M] can refer to either the deck in general, or the distance of a specific finger from the zero'th gap. Examples:

A) I dropped an ACT[M] deck on the floor.
B) The ACT[M] value for the 5.5'th finger is .54E-3 (i.e. ACT[5.5]=.54E-3, or, the finger numberd 5.5 is a distance of .54E-3 from the center of the zero'th gap).

"W3[M]" gives the distance between the main (i.e. electrically active) and dummy elements. W3[M] is almost always constant within a comb. If the W3[M] value is left off the card, the program will use the W3 value from namelist $CONVERT. W3[M] is ignored if dummy elements are not being used (see Figures 3.1, 3.2, 3.3, 3.4).

3.4.4 Dummy Elements, Single, Double Electrodes

A dummy element is an electrically inactaive element that goes directly opposite (separated by distance W3[M]) the electrically active element. Dummy elements are not always used. If DUMMY = .True. on the $CONVERT namelist then dummy elements are included. If DUMMY = .False.then dummy elements are not included. Dummy fingers are always the same width as the main finger they are opposite.

Similarly, if SINGLE = .True. on the $CONVERT namelist then the program uses single finger elements. If SINGLE = .False. then the program uses double finger elements (see Figures 3.1, 3.2, 3.3 and 3.4).

3.4.5 Deck Setup, Defaults

The Ho[N] and ACT[M] decks do not both have to be included. Either one (but not both) of the decks may be left out.

If the Ho[N] deck is left out, the "N" numbers can be calculated from the "M" numbers on the ACT[M] deck. Example: The M numbers may go: (-3.5, -2.5, -1.5, -.5, 15, 1.5, 2.5, 3.5). In this case the N numbers would be calculated as (-3, -2, -1, 0, 1, 2, 3). The Ho[N] values would all be assumed to be 1 (their real world value being "OVALAP", from namelist $CONVERT).

If the ACT[M] deck is left out, the "M" numbers can be calculated from the "N" numbers on the Ho[N] deck. Example: The N numbers may go (-3, -2, -1, 0, 1, 2, 3). In this case the M numbers would be calculated as (-3.5, -2.5, -1.5, -.5, .5, 1.5, 2.5, 3.5). The ACT[M] values would be calculated by assuming the gaps between elements to be equal to W1. (Although W1 and W2 have different definitions depending on whether they are describing a comb with single or double electrodes, in this particular case, with the ACT[M] deck left out, the gaps between all fingers have a distance of W1.)

Thus for single finger electrodes ACT[M] is calculated by the formula:

$$ACT[M] = (2)(M)(W1)$$

For double finger electrodes ACT[M] is calculated by the formula:

$$ACT[M] = 2M(W1 + W2)$$

(See Figures 3.5, 3.6)

When the ACT[M] deck is missing W1[M], W2[M] and W3[M] are all obtained from the $CONVERT namelist.

i.e.	W1[M] = W1	For all M. (W1, W2, W3
	W2[M] = W2	values from
	W3[M] = W3	namelist $CONVERT.)

3.4.6 Miscellaneous Points of Interest

3.4.6.1 ACT[M] Input Format

The format to read in the ACT[M] title card is (8A10).

The format to read all other ACT[M] cards is F6.1, 4E15.8.

3.4.6.2 Connecting Electrode to Correct Pad

How does one know if a particular electrode is connected to the top or bottom pad? The convention is, that the first electrode on the positive side of the zero'th gap is connected to the bottom pad. All other finger connections can then be figured out from the algorithms given in this section.

3.4.6.3 Reversal of Combs in Multi-Comb Device

When CONVERT handles a two comb device it reverses the first comb. In other words, if we were to input two identical combs into program CONVERT, the output would be two combs (described in Raytheon format), the first comb being the mirror image of the second. In a three comb device with NCOMBS=3 (from the $CONVERT namelist) no combs are reversed. If NCOMBS = -3, the first comb is reversed. In a four comb device the first and second combs are reversed. This is shown graphically in Figure 3.7.

3.4.6.4 The Distance "PADS"

The elements on either side of the zero'th gap are always the same length, that is, the element M = 1/2 is the same length as element M = -1/2. The distance of the 1/2'th element from the top Pad is "PADS" by implication; the distance of the -1/2'th element from the bottom Pad is also "PADS". If this were not the case there would not be enough information for "CONVERT" to convert the RADC formats to the Raytheon format (see Figures 3.1, 3.2, 3.3, 3.4).

3.4.6.5 T's in Column 80 of Ho[N] Deck

When a T appears in Column 80 of one of the Ho[N] cards, then the left most electrode surrounding this gap is removed (or thinned). The

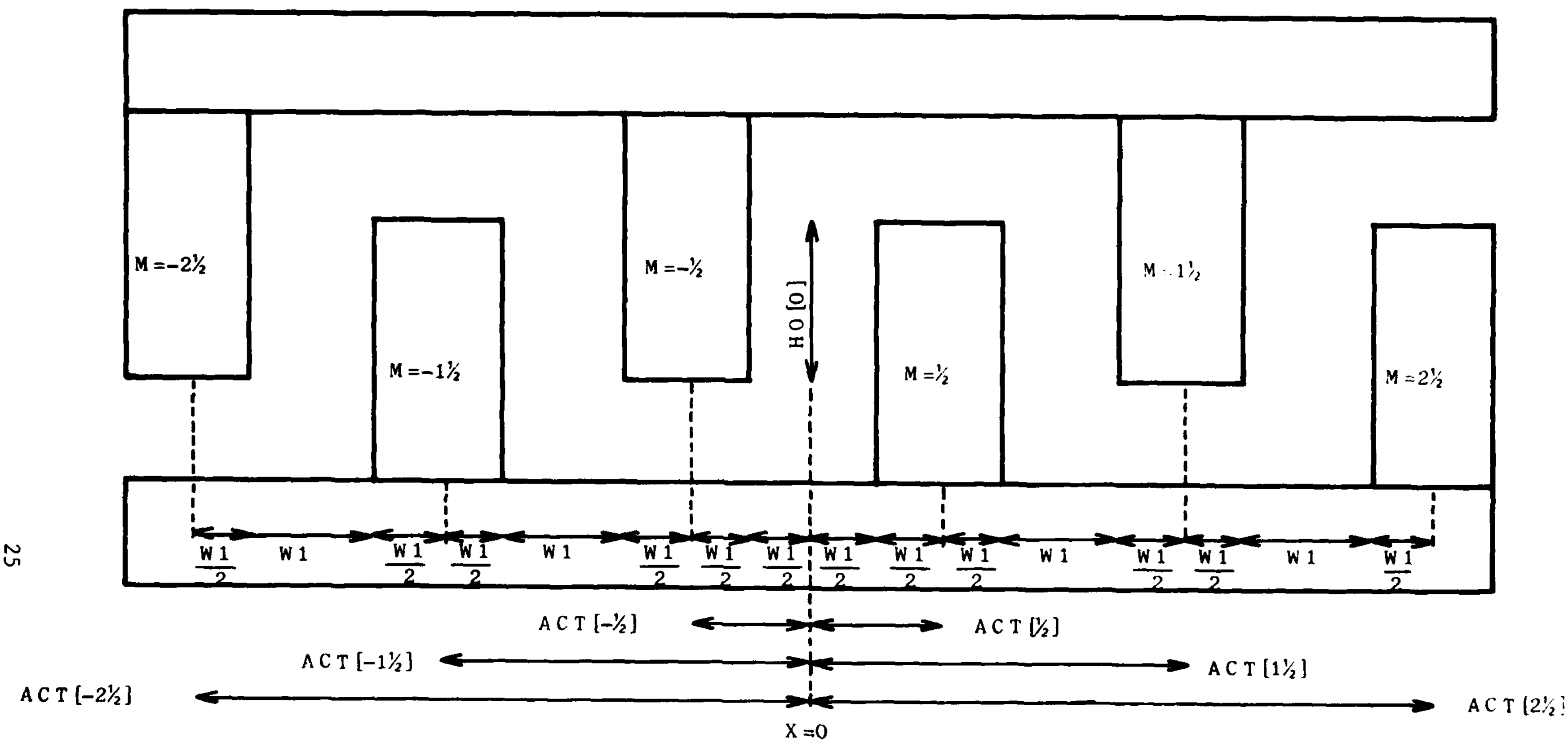


Figure 3.5 A Single Electrode Comb

W1[M] is defined as the width of the M'th finger. When the ACT[M] deck is not included then W1[M] = W1, for all M, the value of W1 being obtained from the $CONVERT namelist. When the ACT[M] deck is not included, the gap between elements is also assumed to be W1. Thus

$$ACT[M] = 2(M)(W1)$$

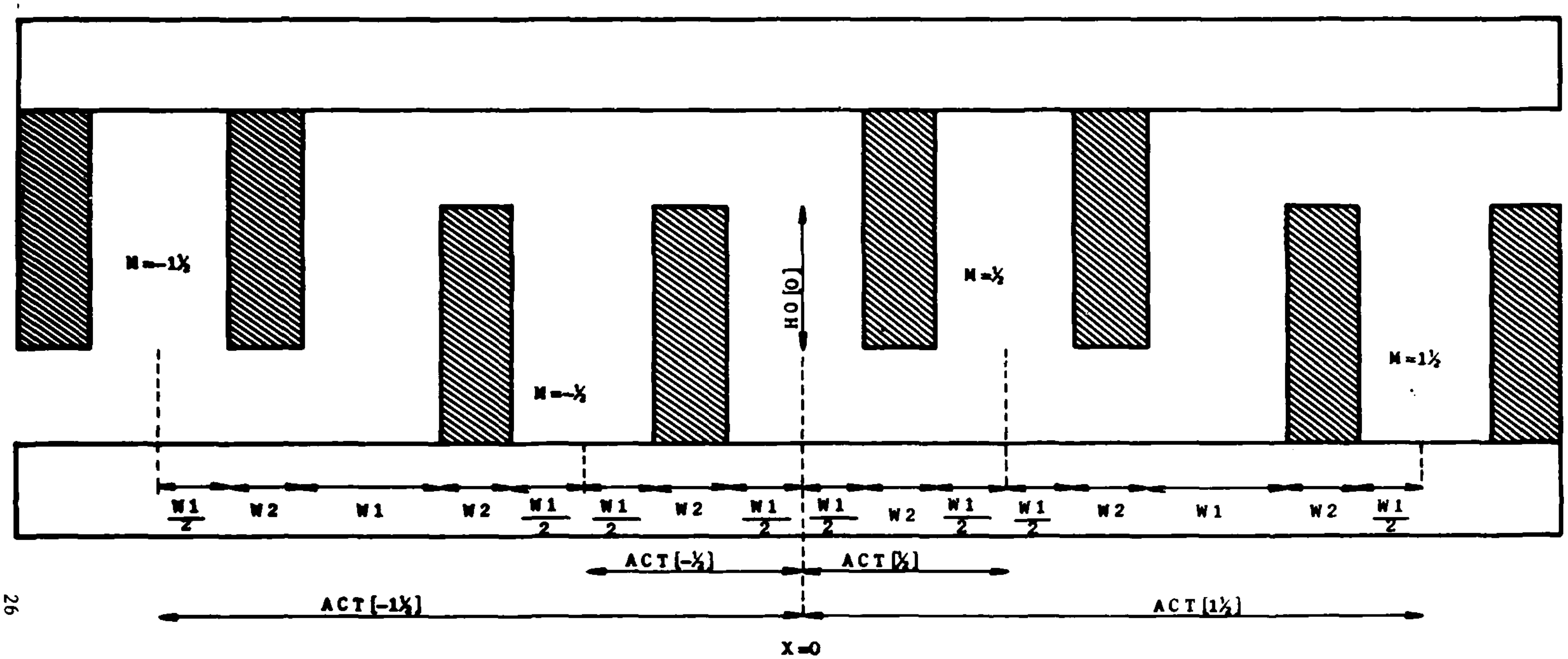


Figure 3.6 A Double Electrode Comb

W1[M] is defined as the gap between the fingers of the M'th double electrode pair. W2[M] is defined as the width of the individual fingers of the M'th double electrode pair. When the ACT[M] deck is not included, then W1[M] = W1 and W2[M] = W2, for all M, the values of W1 and W2 being obtained from the $CONVERT namelist. When the ACT[M] deck is not included the gap between double electrode pairs is also assumed to be W1. Thus

$$ACT[M] = 2M(W1 + W2)$$

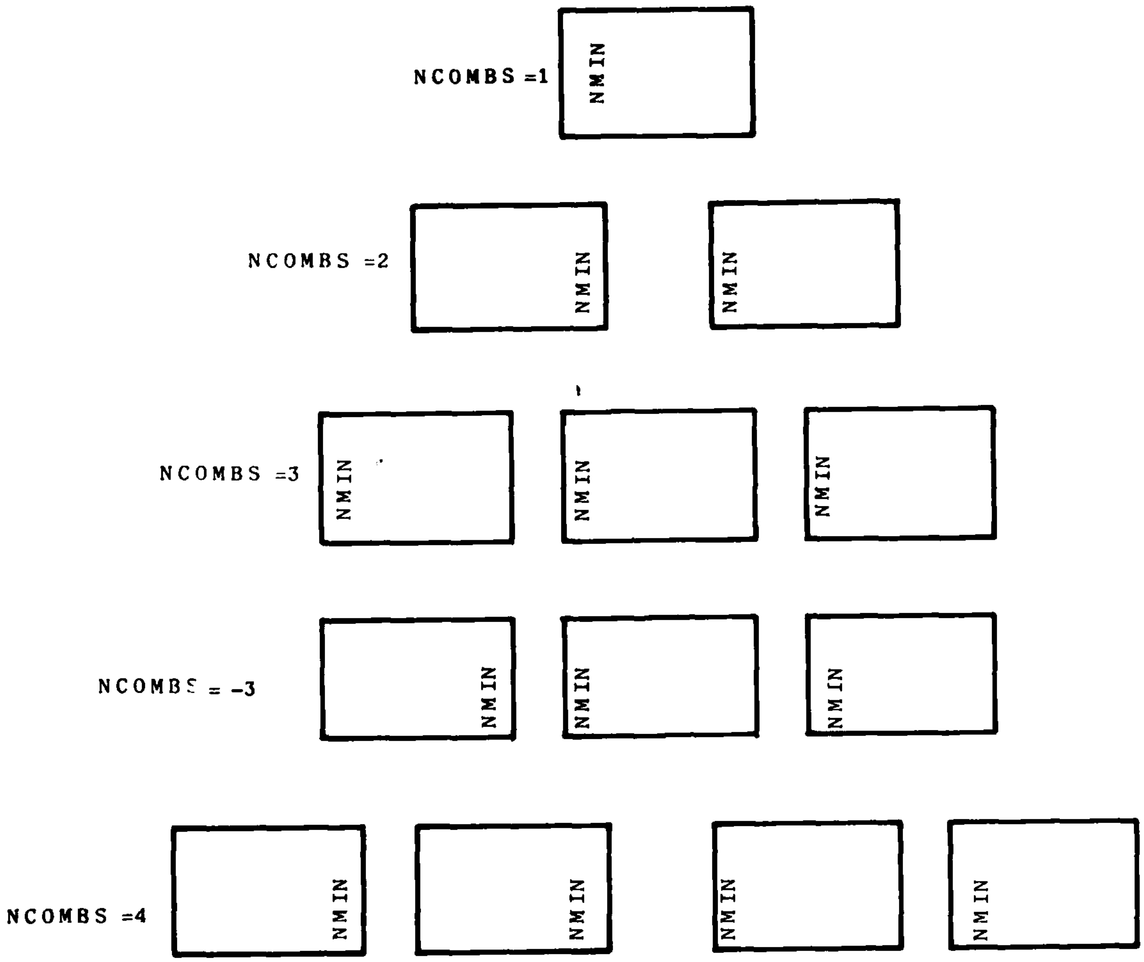


Figure 3.7 Certain Combs Reversed In Multi Comb Devices.

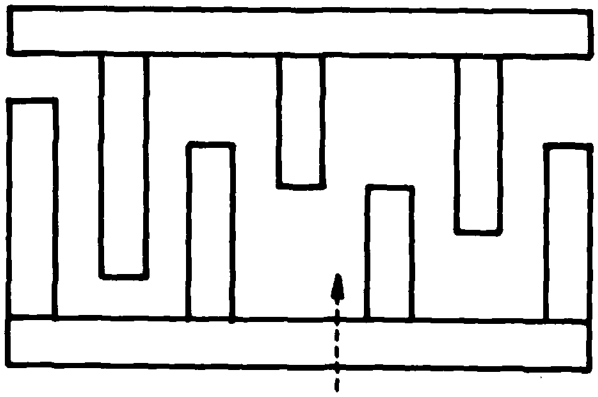


Figure 3.8 A Fictitious Zero Gap

This is not an example of a zero gap. This situation never occurs.

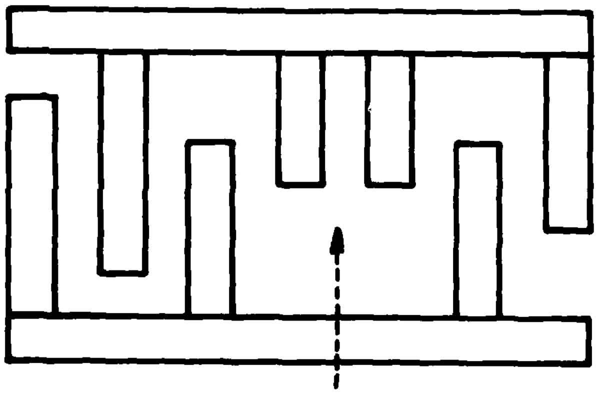


Figure 3.9 Example of Zero Gap

If the Ho[N] values to the left of the zero gap are positive then the Ho[N] values to the right will be negative and vice versa.

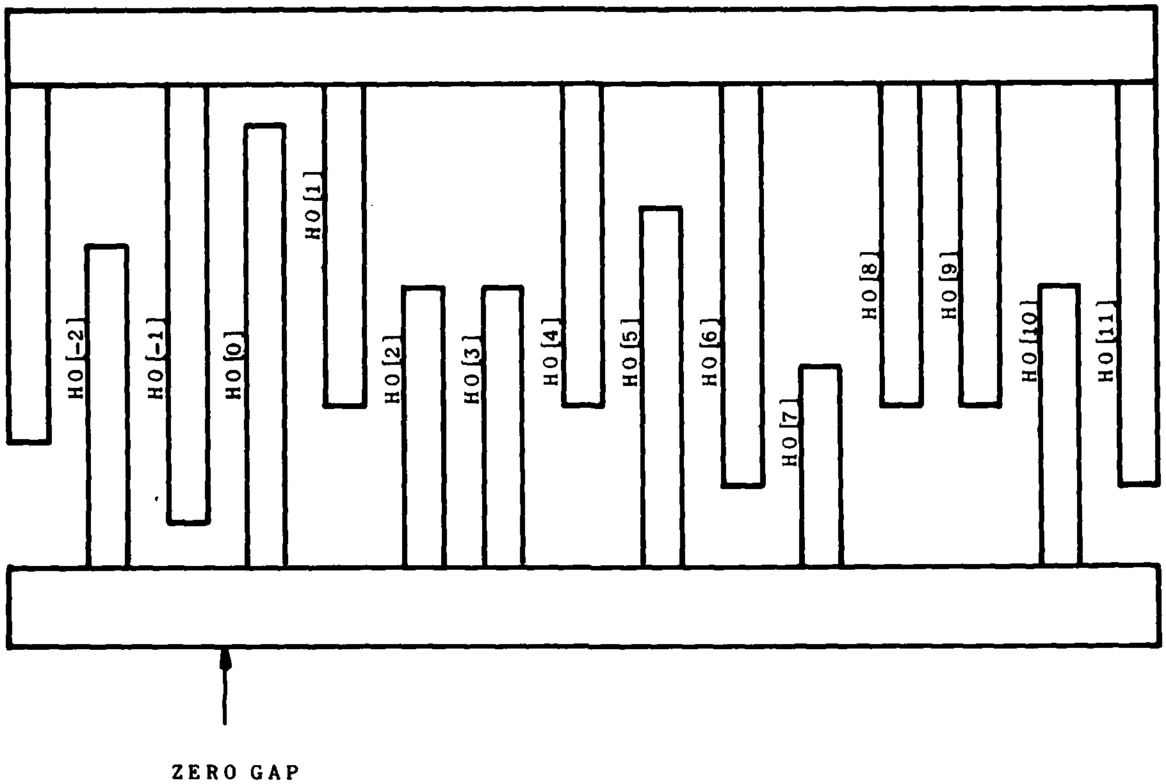


Figure 3.10 Transducer with Negative Ho[N] Values

$H_0[-2]$	= .5	$H_0[5]$	= -.5
$H_0[-1]$	= .7	$H_0[6]$	= -.7
$H_0[0]$	= 1.0	$H_0[7]$	= -.3
$H_0[1]$	= .7	$H_0[8]$	= -.1
$H_0[2]$	= .3	$H_0[9]$	= 0
$H_0[3]$	= 0	$H_0[10]$	= .3
$H_0[4]$	= -.3	$H_0[11]$	= .5

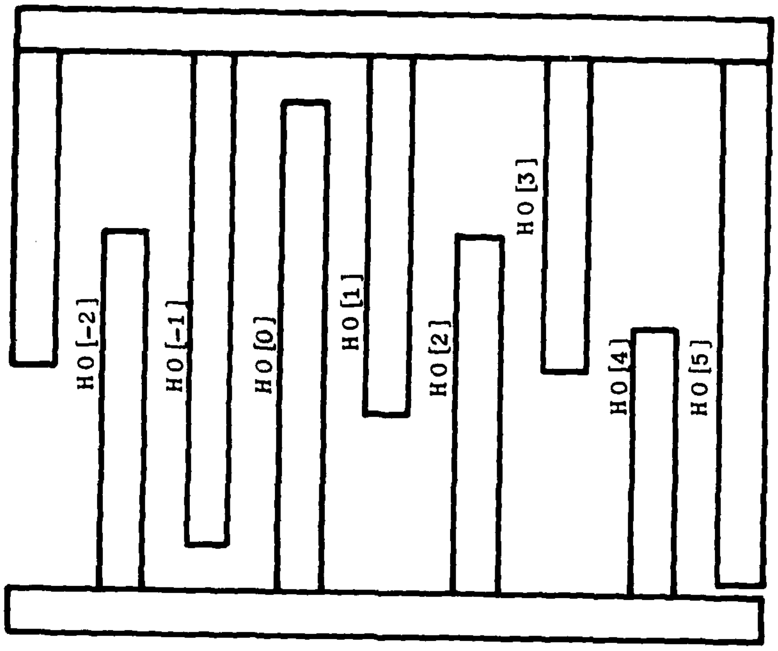


Figure 3.11 Pathological Case of Impossible $H_0[N]$ Value

$H_0[-2] = .3$
$H_0[-1] = .7$
$H_0[0] = 1$
$H_0[1] = .7$
$H_0[2] = .4$
$H_0[3] = .3$
$H_0[4] = .1$
$H_0[5] = .8$

As one can see from the diagram, the maximum possible $H_0[5]$ value is slightly less than .6. $H_0[5] = .8$ is impossible.

rest of the comb will remain the same (i.e. spacing is preserved). This is a seldom used option.

3.4.6.6 Device Description Units

Devices are always specified in units of meters.

3.4.6.7 Discussion of RADC Standard Format

The RADC standard format is quite useful from the viewpoint of the design engineer.

The Ho[N] array not only defines the element overlaps, but also the impulse response of the comb being defined, i.e. the shape of the surface wave launched from the comb as a result of a pulse applied to the input port of the comb. Also, in many cases, it is desirable to place the elements in a comb non-periodically in order to compensate for second order effects such as surface wave diffraction. In these cases, the design engineer would specify ACT[M] as some mathematical function of M, and then create an ACT[M] deck accordingly.

Ho[N] and ACT[M] are looked upon as functions related to the device characteristics rather than specifications of physical appearance.

3.4.6.8 Maximum Deck Size

The maximum number of Ho[N] cards allowed for each comb is 999; the maximum number of ACT[M] cards allowed for each comb is 1000.

3.5 RADC Coded Format

3.5.1 General Description of a Coded Transducer

The coded format arose as a simplification of the description of tapped or thinned transducers. It is used with investigations into corrections of effects of surface wave diffraction (both amplitude and phase) in interdigital transducers.

This kind of transducer is illustrated in Figures 3.12, 3.13, 3.14 and 3.15. A coded transducer consists of a number of groups of electrodes. Each group of electrodes is referred to as a tap.

Within a tap the element overlap Ho[NTAP] is always constant. However, the element overlap may vary from tap to tap. The largest element overlap should always occur in the zero'th tap. The element overlap in other taps may be as large, but should never be larger than the element overlap in the zero'th tap. The element overlap of the zero'th tap should always be 1. Element overlaps in other taps are scaled accordingly. The real value of the element overlaps can be found by multiplying the scaled value by the value "OVALAP" from the $CONVERT namelist.

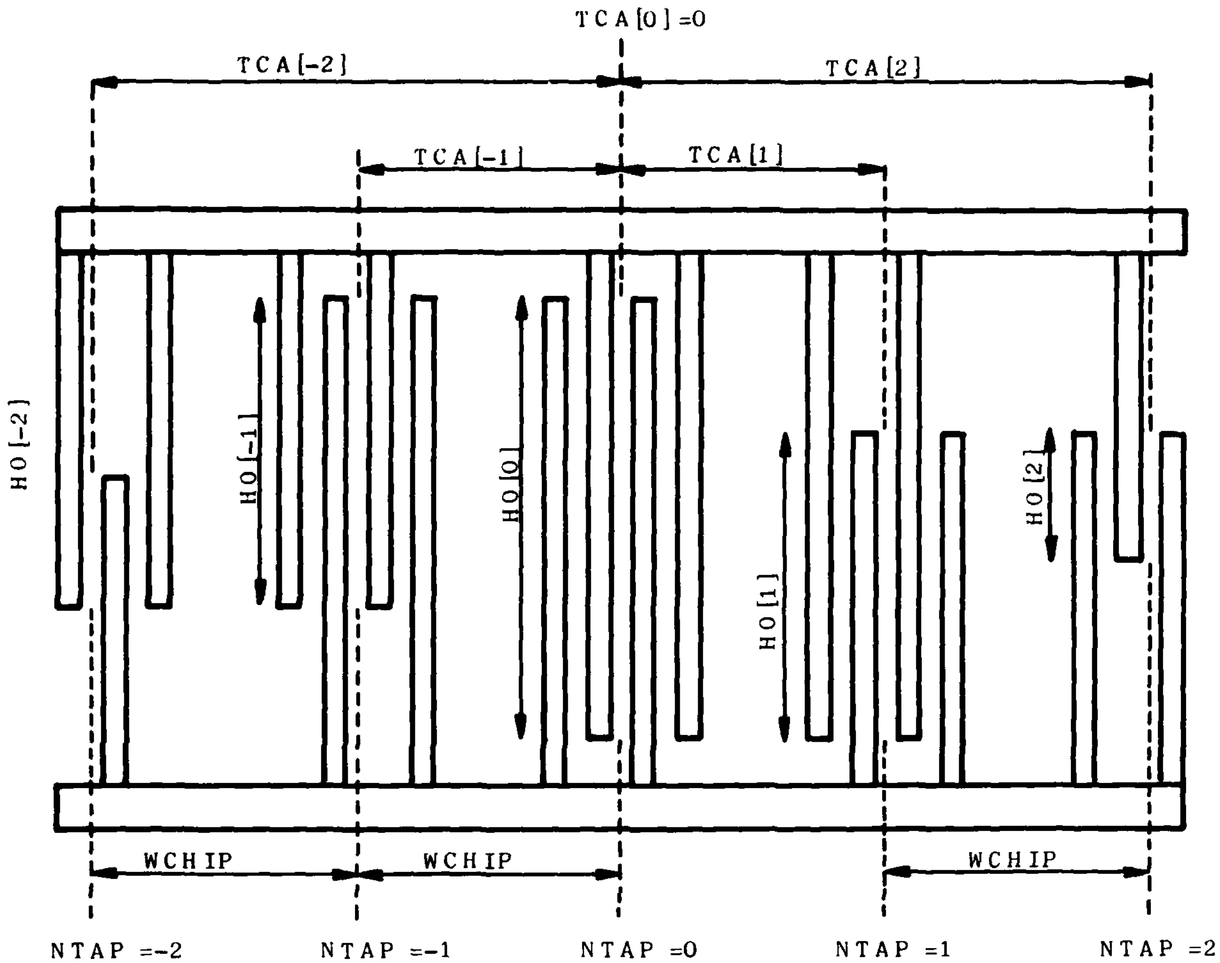


Figure 3.12 Coded Transducer, Odd Number of Taps.
<u>Even Number of Fingers per Tap (Normal Case)</u>

Assuming Realistic Distances along X-axis, then

$H_0[-1]$	=	.3	TCA[-2]	=	-12E-6
$H_0[-1]$	=	.7	TCA[-1]	=	-6E-6
$H_0[0]$	=	1	TCA[0]	=	0
$H_0[1]$	=	.7	TCA[1]	=	6E-6
$H_0[2]$	=	.3	TCA[2]	=	12E-6

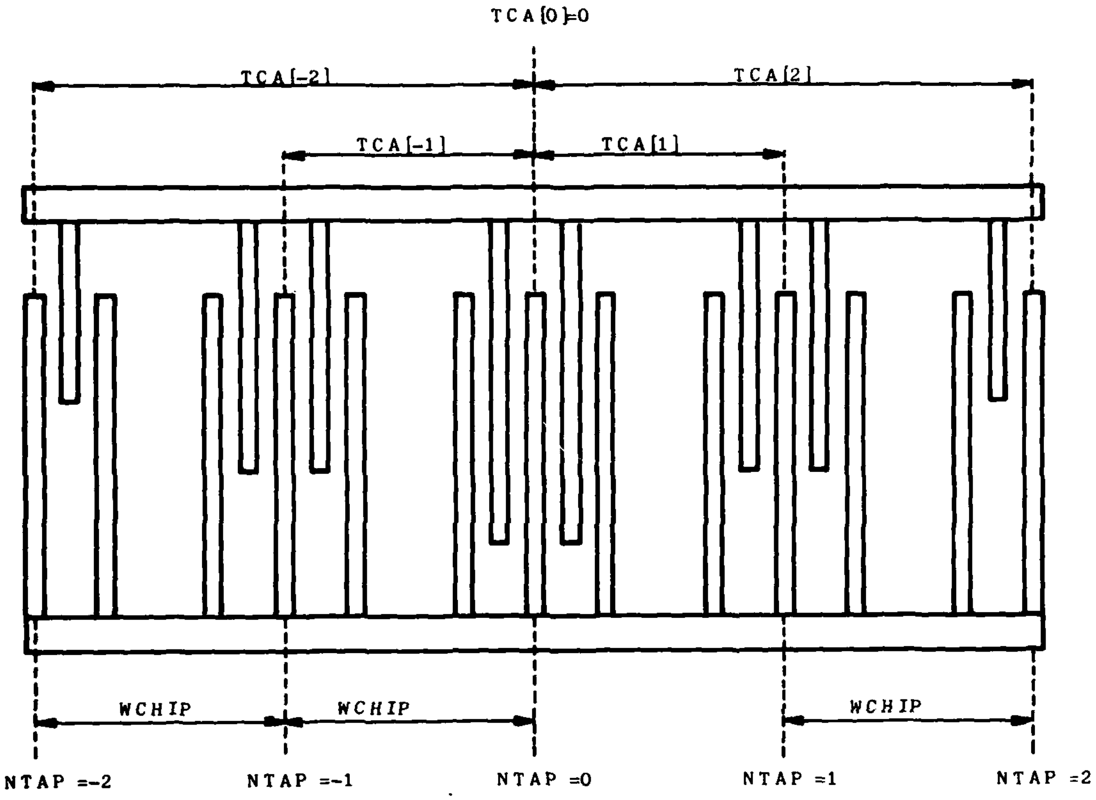


Figure 3.13 Coded Transducer, Odd Number of Taps, Odd Number of Fingers per Tap

Assuming Realistic Distances along X-axis, then

$H_0[-2]$	=	.428	TCA[-2]	=	-14E-6
$H_0[-1]$	=	.714	TCA[-1]	=	-7E-6
$H_0[0]$	=	1	TCA[0]	=	0
$H_0[1]$	=	.714	TCA[1]	=	7E-6
$H_0[2]$	=	.428	TCA[2]	=	14E-6

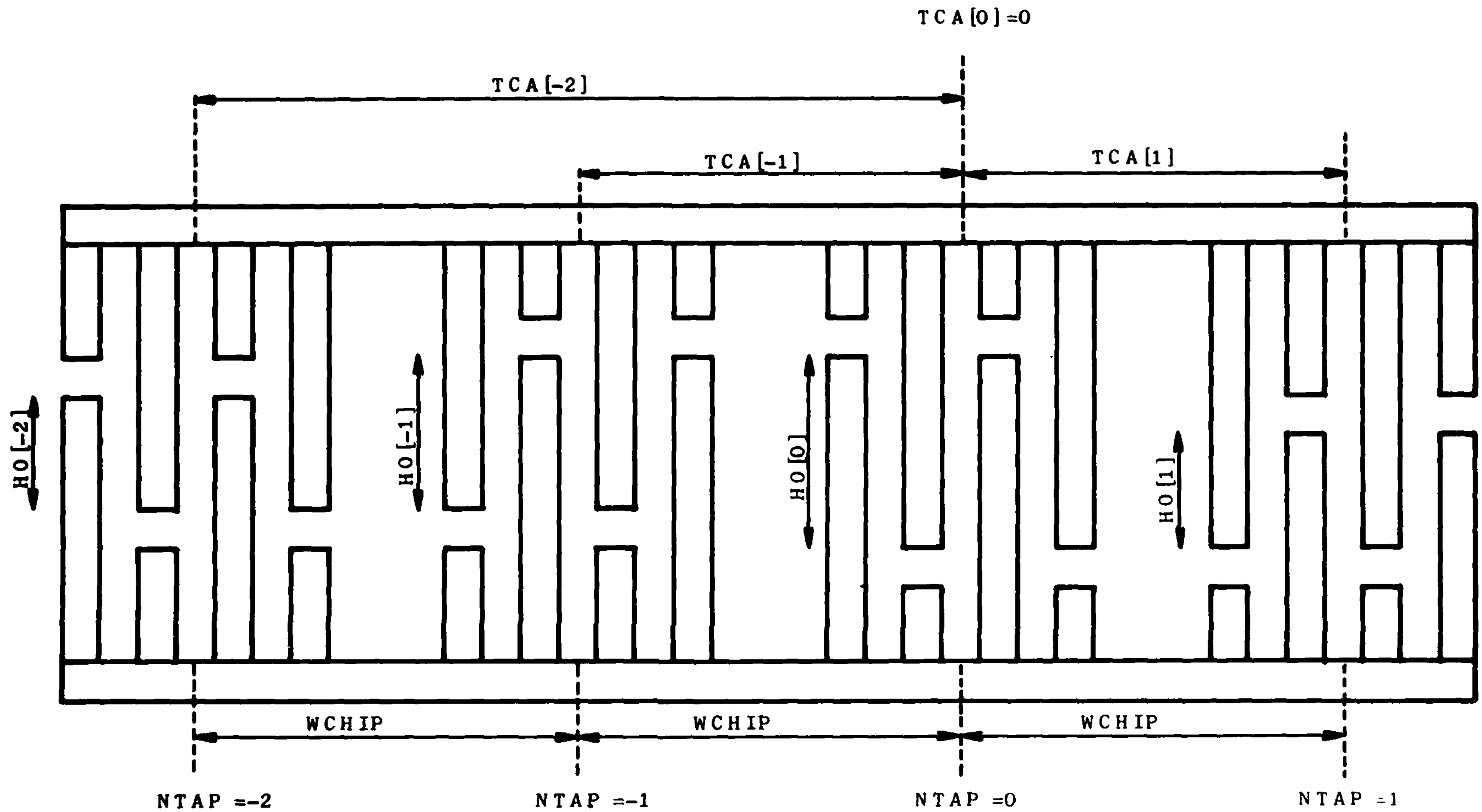


Figure 3.14 Coded Transducer with Dummy Fingers, Even Number of Taps, Even Number of Fingers per Tap

Assuming Units of Realistic Distances along X-axis, then

$H_0[-2]$	= .6	TCA[-2]	= -20E-6
$H_0[-1]$	= .8	TCA[-1]	= -10E-6
$H_0[0]$	= 1	TCA[0]	= 0
$H_0[1]$	= .6	TCA[1]	= 10E-6

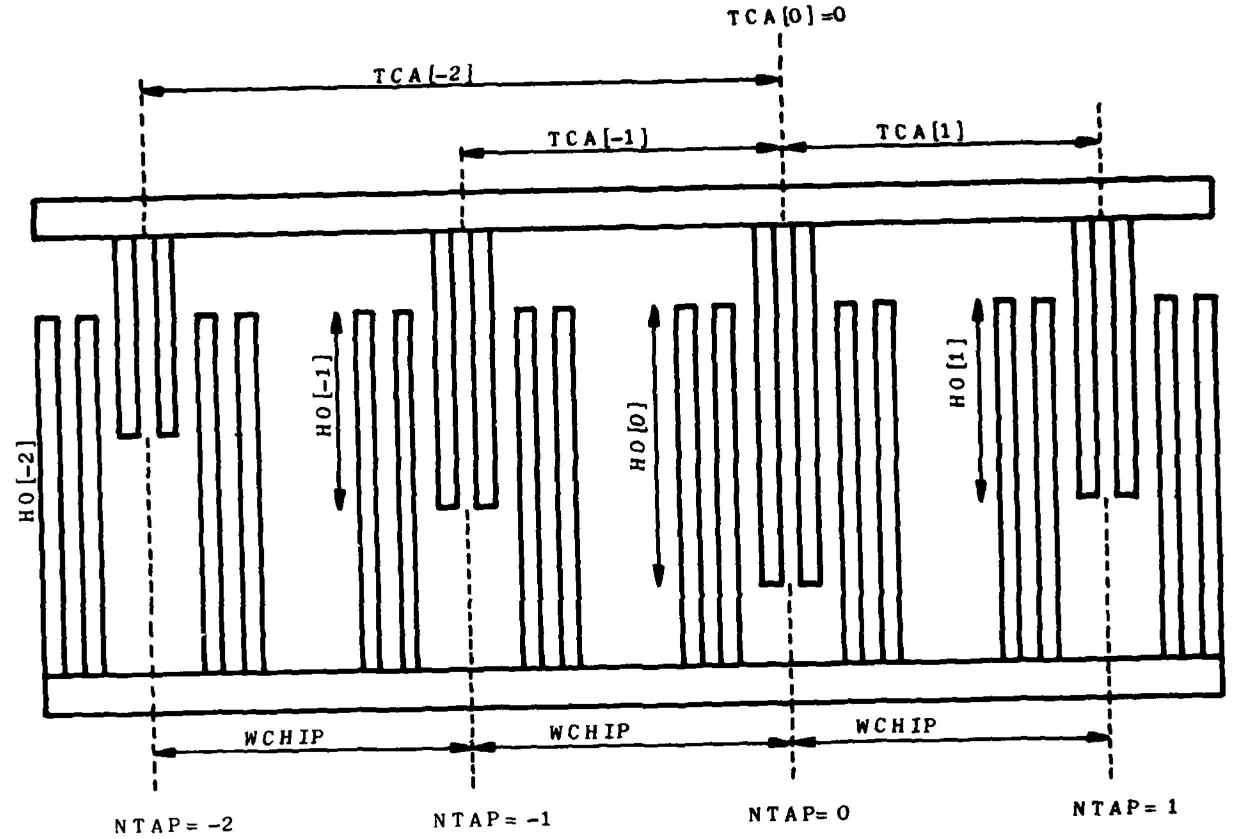


Figure 3.15 Coded Transducer with Double Electrodes, Even Number of Taps, Odd Number of Fingers per Tap

Assuming Units of Realistic Distances along X-axis, then

$H_0[-2]$	=	.428	TCA[-2]	=	-16E-6
$H_0[-1]$	=	.714	TCA[-1]	=	-8E-6
$H_0[0]$	=	1	TCA[0]	=	0
$H_0[1]$	=	.714	TCA[1]	=	8E-6

A coded transducer may consist of single electrodes without dummy fingers, single electrodes with dummy fingers, double electrodes without dummy fingers or double electrodes with dummy fingers. The type of element used must, however, be constant within the whole comb (i.e. element types may not vary from tap to tap).

With the exception of the end taps, all taps within a comb contain the same number of elements, this number being "NF" from the $CONVERT namelist.

The leftmost tap and the rightmost tap (referred to as the -NTAPMIN'th tap and NTAPMAX'th tap respectively) have the same number of elements as each other, this number being calculated in one of three possible ways: (The only exception to this rule is when the $CONVERT namelist variable ADD is not equal to zero, see Section 3.6)

A) If the number of taps in the comb is even, then number of elements in end taps = NF (see Figures 3.14, 3.15).

B) If the number of taps in the comb is odd and "NF" is even, then number of elements in end taps = NF/2 + 1 (see Figure 3.12).

C) If the number of taps in the comb is odd and "NF" is odd, then number of elements in end taps = (NF+1)/2 (see Figure 3.13).

For various calculations (such as the placement of the taps) it is necessary to know where the so-called "center" of the tap is*. With the exception of the end taps the so-called center is the same as the physical center. For non-end taps with "NF" odd, the center of the tap is at the center of the middle element (see Figures 3.13, 3.15). For non-end taps with "NF" even, the center of the tap is at the center of the middle gap** (see Figures 3.12, 3.14).

For Case A (as described above, Figures 3.14, 3.15) the end taps are identical to the other taps and so is their center point.

For Case B (Figure 3.12) the so-called "center" of the left end tap is at the center of the leftmost gap. The so-called "center" of the right end tap is at the center of the rightmost gap.

For Case C (Figure 3.13) the so-called "center" of the left end tap is at the center of the leftmost element. The so-called center of the right end tap is at the center of the rightmost element.

---

*We refer to the horizontal (and not the vertical) center of the tap (i.e. the center of the tap along the X-axis).

**For this discussion "gap" refers to the distance between elements, and not to the distance between fingers of a double electrode.

To figure out which elements are connected to which bus bars the following algorithm is used:

A) The leftmost element of the zero'th tap is connected to the bottom bus bar.

B) Within a tap the elements alternate between being connected to the top and bottom bus bars.

C) The rightmost element of any tap is connected to the same bus bar as the leftmost element of the tap on its right. Conversely, the leftmost element of any tap is connected to the same bus bar as the rightmost element of the tap on its left

Also note that the rightmost element of any tap and the leftmost element of the tap on its right are always the same length. Conversely, the leftmost element of any tap and the rightmost element of the tap on its left are always the same length (see Figures 3.12, 3.13, 3.14, 3.15).

For both single and double electrodes, the distance between elements of a tap is W1 (even though W1 does not have the same meaning for both single and double electrodes). (See Section 3.4.3.)

3.5.2 RADC Coded Format Setup

The RADC coded format consists of a namelist $CONVERT and two decks for each comb to be described by this format. The first deck is referred to as the Ho[NTAP] deck (pronounced H zero of N tap). The second deck is referred to as the TCA[NTAP] deck. Both these decks are optional as will be described in the following sections.

3.5.3 The Ho[NTAP] Deck

The Ho[NTAP]deck consists of a title card and a card for each tap of the comb. Each card following the title card consists of the number, NTAP, (an integer) of the tap and Ho[NTAP] (real), the amount of element overlap in the NTAP'th tap. The taps may be numbered from left to right, or from right to left, the restrictions being:

A) They must be numbered consecutively.

B) One of the taps must be labeled Tap Zero

C) The zero'th tap must contain the longest element overlaps.

Other taps may have equal element overlaps, but none can have more.

From these conditions it is easy to see that some of the taps must be numbered negatively. The tap numbered most negatively is called the -NTAPMIN'th tap. The number "NTAPMIN" is always positive. The tap labeled with the highest positive number is called the NTAPMAX'th tap. The number NTAPMAX is always positive.

When a comb has more then one tap with maximum overlap (often all the overlaps are equal), then it is up to the program user to decide which tap he wants to label zero.

As described in Section 3.5.1, the overlap value on the card for the zero'th tap is always 1. The other element overlap values are scaled accordingly. The actual overlap of the zero'th tap is "OVALAP", defined by the $CONVERT namelist. Thus, the actual overlap of the elements in the NTAP'th tap is simply (OVALAP) (Ho[NTAP]).

The format of the Ho[N] title card is (8A10); the format of the cards following the title card is (I5, E15.8).

3.5.3.1 Negative Ho[NTAP] Values, Zero Ho[NTAP] Values and Almost Zero Ho[NTAP] Values

Just as zero and negative Ho[N] values are possible (see Section 3.4.1) so are zero and negative Ho[NTAP] values possible.

Actually the program is only able to handle negative taps that have an even number of elements. Attempts to produce negative taps with an odd number of elements per tap will produce erroneous results.

Mathematically the sign of the Ho[NTAP] value changes everytime a zero Ho[NTAP] value occurs. That is, if Ho[NTAP] = 0, then the signs of Ho[NTAP-1] and Ho[NTAP+1] will be opposite.

A zero tap is basically the same as a normal tap except that the fingers do not overlap. The vertical distance of two opposite elements from each other is the scaled distance W3/OVALAP (see Figure 3.16).

A change of sign cannot occur without a zero tap.

If a change of sign does occur in an Ho[NTAP] deck without a zero HO[NTAP]value included, the program will automatically change one of the two Ho[NTAP] values involved in the change of sign, to zero.

The algorithm used to do this is the same algorithm described in Section 3.4.1.

Both parts "A" and "B" apply. The only difference is that N gets replaced by NTAP and NMIN and NMAX get replaced by NTAPMIN and NTAPMAX respectively. All Ho[NTAP] decks are processed by this algorithm.

Step "A" eliminates cases where Ho[NTAP] is so close to zero as to possibly cause problems.

This algorithm only handles the normal cases (see Section 3.4.1).

For a detailed example of a transducer with negative Ho[NTAP] values see Figure 3.17.

3.5.3.2 The Pathalogical Case of Impossible Ho[NTAP] Values

In the following example with NF=4, it would be impossible to implement Ho[4] (see Figure 3.18).

Ho[-1] = .7

Ho[0] = 1.0

Ho[1] = .4

Ho[2] = .4

Ho[3] = .1

Ho[4] = .8

This case will probably never come up, but the user should be aware of its existence. In the event it does come up the program will continue to process incorrectly.

3.5.4 The TCA[NTAP] Deck

The TCA[NTAP] deck consists of a title card and a card for each tap of the comb.

Each card following the title card consists of the numbers NTAP and TCA[NTAP].

As previously described, "NTAP" is the number of the tap.

TCA[NTAP]is the actual (i.e. unscaled) distance from the center of the zero'th tap to the center of the NTAP'th tap. (For a definition of "center" of tap see Section 3.5.1.) TCA[0] always equals zero. If NTAP is negative then the value of TCA[NTAP] is negative (see Figure 3.12, 3.13, 3.14, 3.15).

Unlike ACT[M] decks, TCA[NTAP] decks neither include W1, W2, nor W-3 values. These values have the same meanings as described in Section 3.4.3, but their values are always obtained from the $CONVERT namelist, and remain constant throughout the whole comb.

3.5.5 Deck Defaults

The Ho[NTAP] and TCA[NTAP] decks do not both have to be included. Either one (or both) of the decks may be omitted.

If the Ho[NTAP] deck is omitted, the "NTAP" numbers can be calculated from (and in fact are exactly the same as) the "NTAP" values on the TCA[NTAP] deck.

The Ho[NTAP] values would all be assumed to be 1 (their real world value being "OVALAP", from the $CONVERT namelist).

If the TCA[NTAP] deck is omitted the "NTAP" numbers can be calculated from (and in fact are exactly the same as) the "NTAP" values on the Ho[NTAP] deck.

When the TCA[NTAP] deck is omitted it is assumed that the distance between the center (as defined in Section 3.5.1) of one tap and the center of the next tap is constant throughout the comb. This distance is called "WCHIP" (width of chip) and is defined by the $CONVERT namelist. "WCHIP" is used, only if the TCA[NTAP] deck is omitted.

Thus, when the TCA[NTAP] deck is omitted:

$$\text{TCA[NTAP]} = (\text{NTAP})(\text{WCHIP})$$

(For all NTAP, for either single or double electrodes, with or without dummy fingers.)

When both the Ho[NTAP] and TCA[NTAP] decks are left out the number of taps is determined from the variable "NCHIPS". This variable is on the $CONVERT namelist and is used only in the event that both decks are omitted. "NCHIPS" stands for the number of chips. A chip is the area of the comb from the center of one tap to the center of the next. The number of taps (called NTAPS) is equal to the number of chips plus one.

$$\text{NTAPS} = \text{NCHIPS} + 1$$

The range of NTAP is determined in one of two ways:

A) If NTAPS is even $\text{NTAPMIN} = \frac{\text{NTAPS}}{2}$, $\text{NTAPMAX} = \frac{\text{NTAPS}}{2} - 1$

B) If NTAPS is odd $\text{NTAPMIN} = \frac{\text{NTAPS}}{2} - 1/2$, $\text{NTAPMAX} = \frac{\text{NTAPS}}{2} - 1/2$

NTAP goes from -NTAPMIN to NTAPMAX.

3.5.6 Miscellaneous Points of Interest

3.5.6.1 ACT[M] Input Format

The format to read in the TCA[NTAP] title card is (8A10).

The format to read all other TCA[NTAP] cards is (I5, E15.8).

3.5.6.2 The Distance "PADS"

PADS is the same value as described in Section 3.4.8. In the case of a coded transducer "PADS" is the distance of the leftmost element (in the case of dummy fingers, the distance from the main part of the leftmost element) of the zero'th tap to the bottom of the top bus bar. By implication, the distance from any of the elements in the zero'th tap to the bus bar it is not connected to, is "PADS".

3.5.6.3 Reversal of Combs in Multi-Comb Device

As described in Section 3.4.7, program CONVERT sometimes reverses some of the combs in a device. The same rules for reversal apply no matter what format is used to describe the combs in a device. In fact any comb may be described by any format (i.e. the same format need not be used to describe each comb in a device). Yet in all cases, the same rules of reversal apply (see Figure 3.7).

3.5.6.4 Maximum Deck Size

The maximum number of Ho[NTAP] cards allowed for each comb is 1000; the maximum number of TCA[NTAP]cards allowed for each comb is 1000.

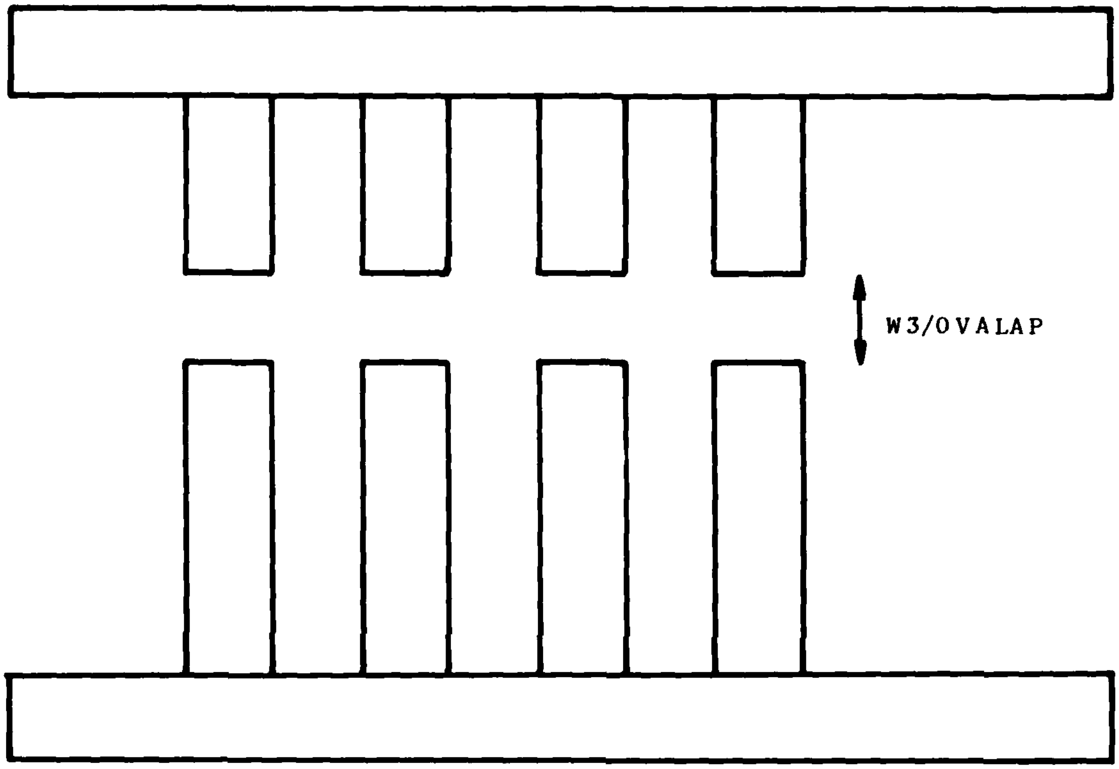


Figure 3.16 A Zero Tap

The distance W3/OVALAP represents a scaled (Raytheon format) distance, the final product would have a gap of W3 meters between the fingers.

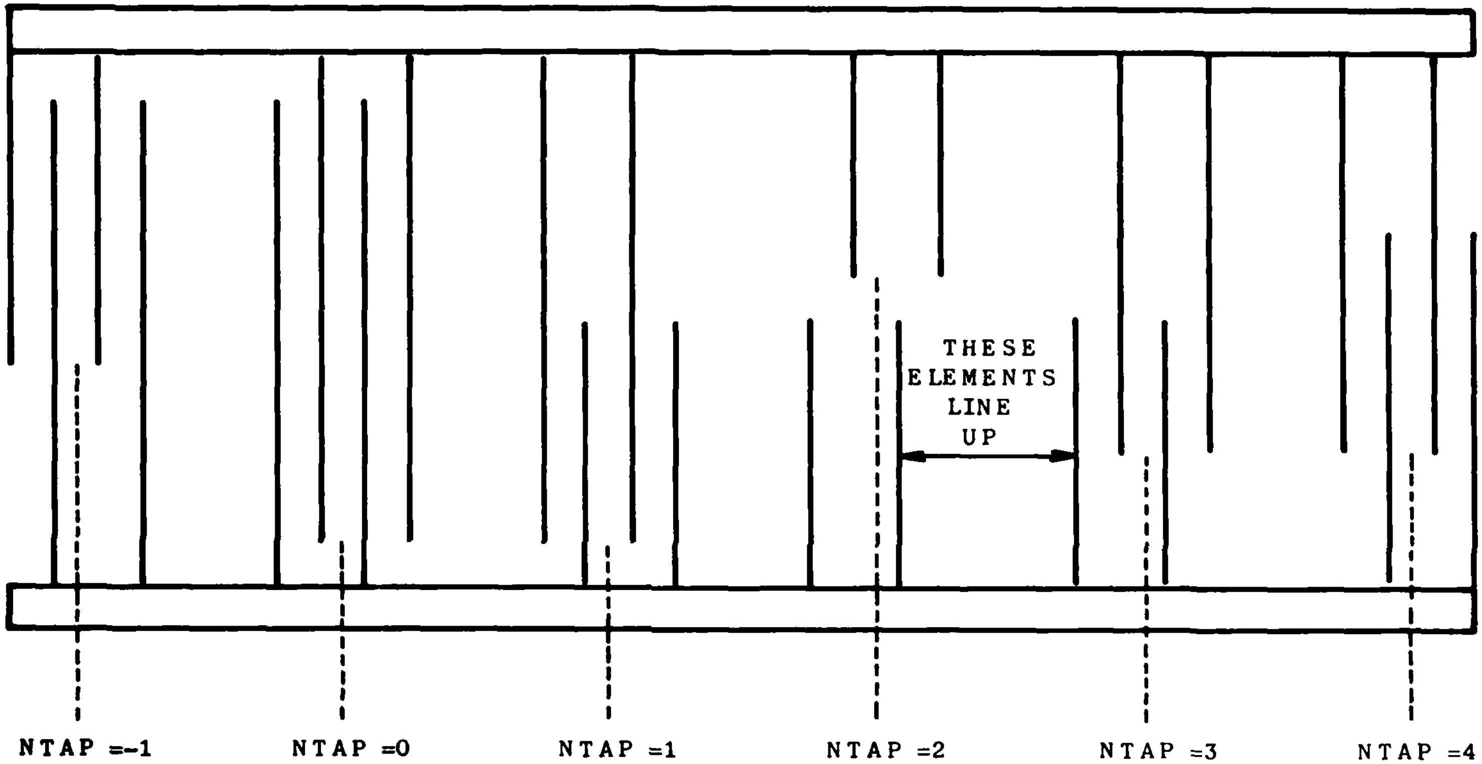


Figure 3.17 Transducer with Negative H_0[NTAP] Values

$H_0[-1]$	=	.6
$H_0[0]$	=	1
$H_0[1]$	=	.5
$H_0[2]$	=	0
$H_0[3]$	=	-.3
$H_0[4]$	=	-.5

Note that after a zero tap, the inside edges of the taps (i.e., the left most element of tap 3 and the right most element of tap 2) do not line up. This always occurs with a zero tap. The right edge of a zero tap (when NTAP is positive) or the left edge of a zero tap (if NTAP is negative) do not line up in the usual way. Instead, the following tap lines up with the second to last element of the zero tap.

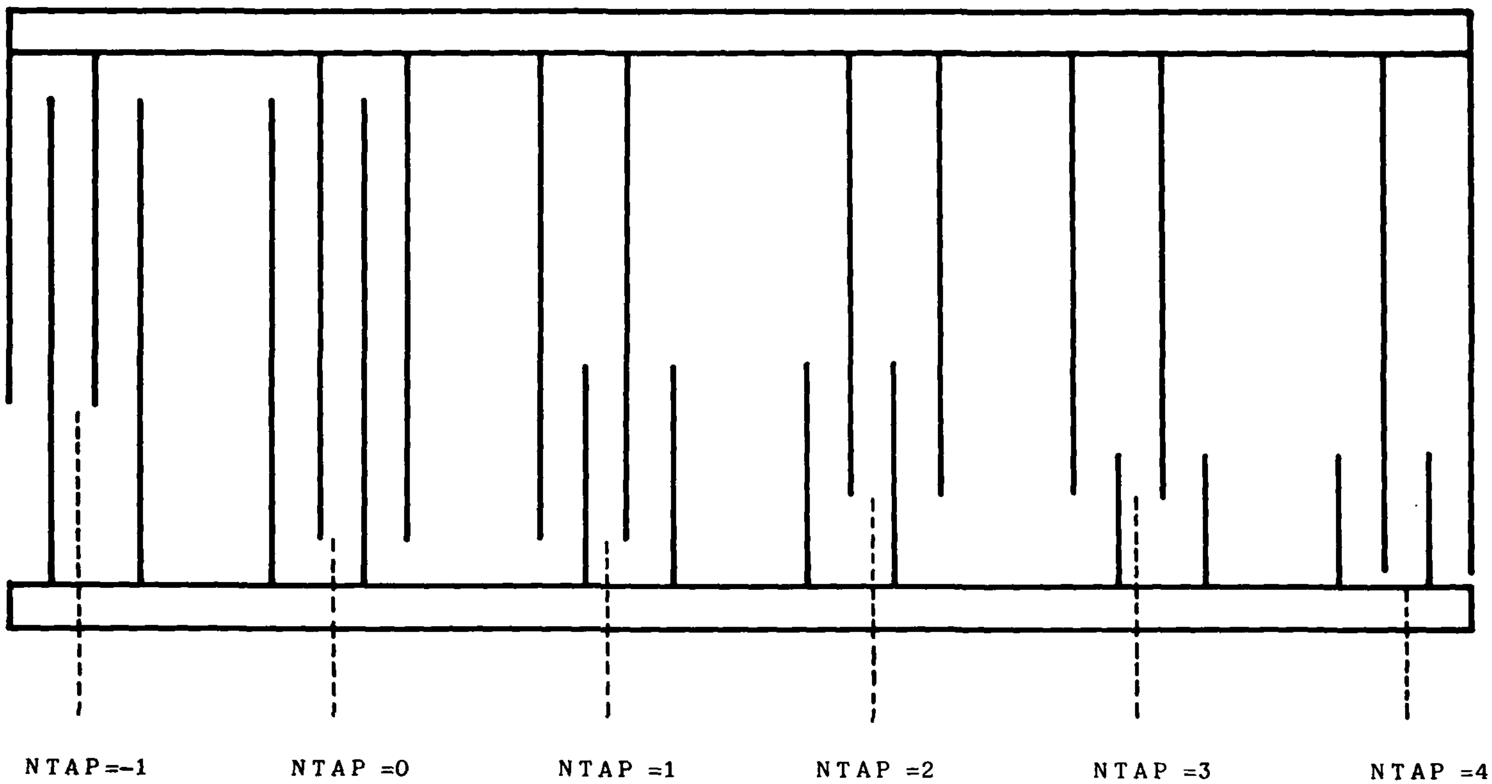


Figure 3.18 Pathological Case of Impossible H_0[NTAP] Value

$H_0[-1]$	=	.7	As one can see from the diagram, the absolutely largest possible $H_0[4]$ value is slightly less than .3. $H_0[4]$ = .8 is impossible.
$H_0[0]$	=	1	
$H_0[1]$	=	.4	
$H_0[2]$	=	.3	
$H_0[3]$	=	.1	
$H_0[4]$	=	.8	

3.6 The $CONVERT NAMELIST

3.6.1 Introduction

The Ho[N] or Ho[NTAP] and ACT[M]or TCA[NTAP] decks do not contain enough information to fully describe a SAW device. To complete the description, the $CONVERT namelist is needed. Since there are up to five combs per device, and each comb is capable of being defined by a different format, most of the variables on the $CONVERT namelist are arrays of length five, the first value of an array variable being used to describe the first comb, the second value used to describe the second comb, etc. The number of combs in the device is given by "NCOMBS", which will determine how many of the five array values are used.

3.6.2 The $CONVERT Namelist Variables and Their Type

All arrays are of length five.

NCOMBS	Integer
OVALAP	Real
PADS	Real
W1	Real Array
W2	Real Array
W3	Real Array
DUMMY	Logical Array
SINGLE	Logical Array
CODE	Logical Array
NCHIPS	Integer Array
NF	Integer Array
WCHIP	Real Array
CARR	Logical Array
LEVER	Logical Array
EDG2EDG	Real Array
AVERAGE	Logical Array
FILL	Logical Array

ADD	Integer Array
ENGAN	Logical Array
FUTURE	Logical Array
EXPAND	Real Array

3.6.3 The $CONVERT Namelist Variables and Their Defaults

NCOMBS	2
OVALAP	500.E-6
PADS	10.E-6
W1	1.2E-6, 1.2E-6, 1.2E-6, 1.2E-6, 1.2E-6
W2	1.2E-6, 1.2E-6, 1.2E-6, 1.2E-6, 1.2E-6
W3	5.0E-6, 5.0E-6, 5.0E-6, 5.0E-6, 5.0E-6
DUMMY	T,T,T,T,T
SINGLE	F,F,F,F,F
CODE	F,F,F,F,F
NCHIPS	0,0,0,0,0
NF	0,0,0,0,0
WCHIP	0,0,0,0,0
CARR	F,F,F,F,F
LEVER	F,F,F,F,F
EDG2EDG	0,0,0,0,0
AVERAGE	F,F,F,F,F
FILL	F,F,F,F,F
ADD	0,0,0,0,0
ENGAN	F,F,F,F,F
FUTURE	F,F,F,F,F
EXPAND	0,0,0,0,0

3.6.4 The $CONVERT Namelist Variables and Their Functions

NCOMBS: Gives the number of combs in a device. NCOMBS can range from 1 to 5.

OVALAP: In the RADC standard format OVALAP gives the actual unscaled overlap of the M=-1/2 and M=1/2 elements. In the RADC coded format OVALAP gives the actual unscaled overlap of elements in the zero'th tap (See Sections 3.4.3, 3.5.1, 3.5.3).

PADS: In the RADC standard format PADS gives both the unscaled distance between the main part of the M=1/2 element and the bottom of the top bus bar, and the distance between the main part of the M=-1/2 element and the top of the bottom bus bar. In the RADC coded format PADS gives the unscaled distance between the main part of any element in the zero'th tap to the bus bar it is not connected to. The variable "EXTRA" is the scaled value of PADS (see Sections 3.4.8, 3.5.6.2, 3.7.2, 3.8.1).

W1(I): In the case of a comb with single electrodes W1 gives the width of the electrodes. In the case of a comb with double electrodes W1 gives the width of the gaps between the fingers of double electrodes.

In the RADC standard format the W1 value from Namelist $CONVERT is used only if the W1[M] values are not included in the ACT[M] deck. In the RADC coded format the value of W1 from the Namelist $CONVERT is always used (See Sections 3.4.3, 3.5.4).

W2(I): In the case of a comb with single electrodes W2 is not used. In the case of a comb with double electrodes W2 is the width of the fingers of the double electrodes. Both fingers of a double electrode are the same width, this width being W2.

In the RADC standard format the W2 value from the $CONVERT Namelist is used only if the W2[M] values are not included in the ACT[M] deck. In the RADC coded format the value of W2 from the $CONVERT Namelist is always used (see Sections 3.4.3, 3.5.4).

W3(I): W3 gives the distance between the main (i.e. electrically active) and dummy elements. W3 is ignored if dummy elements are not being used in the comb.

In the RADC standard format the W3 value from the $CONVERT namelist is used only if the W3[M] values are not included in the ACT[M] deck. In the RADC coded format the value of W3 from the $CONVERT Namelist is always used (see Sections 3.4.3, 3.5.4).

DUMMY(I): If DUMMY(I) = True, then the I'th comb will use dummy fingers.

If DUMMY(I) = False, then the I'th comb will not used dummy fingers.

SINGLE(I): If SINGLE(I) = True, then the I'th comb will use single electrodes.

If SINGLE(I) = False, then the I'th comb will use double electrodes.

AVERAGE(I): If AVERAGE(I) = True, then the Ho[N] values for the I'th comb will be averaged in the following way.

Let Ho[-NMIN-2] = Ho[-NMIN]
Let Ho[-NMIN-1] = Ho[-NMIN]
Let Ho[NMAX+1] = Ho[NMAX]
Let Ho[NMAX+2] = Ho[NMAX]

Then for N = -NMIN to NMAX

If (N is odd) and (Ho[N] >0) and Ho[N+1]>0) then $Ho'[N] = \frac{Ho[N] + Ho[N+1]}{2}$

If (N is odd) and (Ho[N]=0) then Ho'[N] = 0

If (N is odd) and (Ho[N-1] <0) and Ho[N]<0) then $Ho'[N] = \frac{Ho[N-1] + Ho[N]}{2}$

If (N is odd) and (Ho[N]>0) and (Ho[N+1]= 0) then $Ho'[N] = \frac{Ho[N] + Ho[N+2]}{2}$

If (N is odd) and (Ho[N-1]=0) and (Ho[N]<0) then $Ho'[N] = \frac{Ho[N] + Ho[N-2]}{2}$

If (N is even) and (Ho[N-1] >0) and (Ho[N]>0) then $Ho'[N] = \frac{Ho[N-1] + Ho[N]}{2}$

If (N is even) and (Ho[N]=0) then Ho'[N] = 0

If (N is even) and (Ho[N]<0) and (Ho[N+1]<0) then $Ho'[N] = \frac{Ho[N] + Ho[N+1]}{2}$

If (N is even) and (Ho[N] <0) and (Ho[N+2]>0) then $Ho'[N] = \frac{Ho[N] + Ho[N+2]}{2}$

If (N is even) and (Ho[N-2]<0) and (Ho[N]>0) then $Ho'[N] = \frac{Ho[N-2] + Ho[N]}{2}$

Then for N = -NMIN to NMAX

Ho[N] = Ho'[N]

and finally Ho[0] = 1

If the RADC coded format is used and AVERAGE(I) = True, the same algorithm is used except:

NMIN gets replaced by NTAPMIN
NMAX gets replaced by NTAPMAX
N   gets replaced by NTAP.

Note: The Ho[N] or Ho[NTAP] deck is processed by the algorithm described in Section 3.4.1 before it is processed by the above algorithm.

FUTURE(I); Future does not serve any purpose at the present time.

EXPAND(I): Expand does not serve any purpose at the present time.

CODE(I): If CODE(I) = True, then the I'th comb is to be described by the RADC coded format.

If CODE(I) = False, then the I'th comb is to be described by the RADC standard format.

NCHIPS(I): NCHIPS(I) is used only if CODE(I) = True, and both the Ho[NTAP] and TCA[NTAP] decks are omitted. NCHIPS indirectly specifies the number of TAPS in the I'th comb.

NTAPS = NCHIPS+1 (See Section 3.5.5)

WCHIP(I): WCHIP(I) is used only if CODE(I) = True, and the TCA[NTAP] deck is omitted. WCHIP is used to fill in the TCA[NTAP] array.

TCA[NTAP] = (NTAP)(WCHIP) (See Section 3.5.5)

NF(I): NF(I) is used only if CODE(I) = True.

NF(I) specifies the number of fingers in each tap (except the end taps) of the I'th comb. (See Section 3.5.1)

CARR(I): If CODE(I) = True and CARR(I) = True then special reflection reducers invented by P. H. Carr are used on the positive (usually right) side of each tap. The reflection reducer on the positive side of a tap is said to be that tap's corresponding reflection reducer. The following rules apply to Carr reflection reducers.

A) A reflection reducer is a group of elements. Each reflection reducer has the same number of elements as its corresponding tap.

B) The same W1, W2, and W3 values apply to the elements of the reflection reducer as to the elements of its corresponding tap.

C) Each element of a reflection reducer is identical to each other. Each element of a reflection reducer is identical to the nearest element of its corresponding tap (be it single electrode without dummy fingers, single electrode with dummy fingers, double electrode without dummy fingers, or double electrode with dummy fingers).

D) The main part of each element of a reflection reducer is connected to the same bus bar. The main part of each element of a reflection reducer is connected to the same bus bar as the main part of the closest element of the corresponding tap.

E) Each reflection reducer on a comb is separated from its corresponding tap by the distance

(EDG2EDG) - (Finger Width)

EDGE2EDGE is on the $CONVERT namelist.

Finger width is the actual width of the fingers (not elements) used in the comb.

If single fingers are used, Finger Width = W1

If double fingers are used, Finger Width = W2

In a comb with the reflection reducer to the right of its corresponding tap the distance (EDGE2EDG - Finger Width) is the distance from the right edge of the rightmost finger of the tap to the left edge of the leftmost finger of the corresponding reflection reducer.

In a comb with the reflection reducer to the left of its corresponding tap the distance (EDGE2EDG - Finger Width) is the distance from the right edge of the rightmost finger of the reflection reducer to the left edge of the leftmost finger of the corresponding tap (see Figure 3.19).

LEVER(I): If CODE(I) = True and LEVER(I) = True
Then special reflection reducers, similar to, but not the same as Carr reflection reducers, are used. These reflection reducers consist of a group of elements on each side of every tap. (In the space

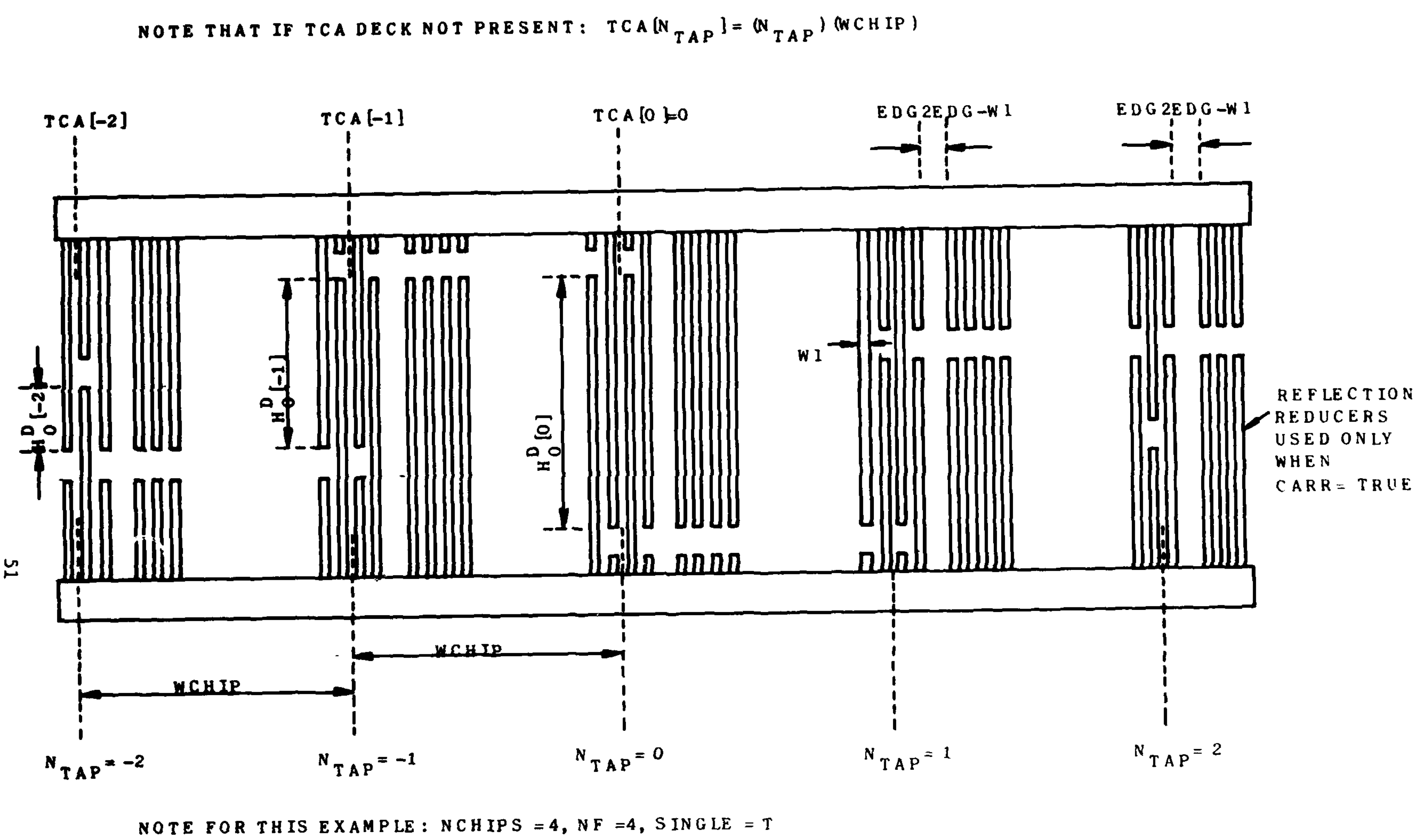


Figure 3.19 Coded Transducer with CARR = True

between two taps there will be two sets of reflection reducers.) The following rules apply to Lever reflection reducers.

A) If the number of elements in a tap is even then the number of elements in each of the two reflection reducers surrounding the tap will be

$$\frac{\text{The Number of Elements in TAP}}{2}$$

If the number of elements in a tap is odd then the number of elements in each of the two reflection reducers surrounding the tap will be

$$\frac{\text{(The Number of Elements in TAP)-1}}{2}$$

B) The same W1, W2, and W3 values apply to the elements of the reflection reducers as to the elements of the corresponding tap.

C) The elements of the reflection reducer on the left side of a tap are identical to each other. Each element of the reflection reducer on the left side of a tap is identical to the leftmost element of that tap. The elements of the reflection reducer on the right side of a tap are identical to each other. Each element of the reflection reducer on the right side of a tap is identical to the rightmost element of that tap (be it single electrode without dummy fingers, single electrode with dummy fingers, double electrode without dummy fingers, double electrode with dummy fingers).

D) The main part of each element of the reflection reducer on the left side of a tap is connected to the same bus bar. The main part of each element of the reflection reducer on the left side of a tap is connected to the same bus bar as the leftmost finger of that tap. The main part of each element of the reflection reducer on the right side of a tap is connected to the same bus bar. The main part of each element of the reflection reducer on the right side of a tap is connected to the same bus bar as the rightmost finger of that tap.

E) Each reflection reducer is separated from its corresponding tap by the distance

EDG2EDG - (Finger Width) (See Part E of Carr Description) (See Figure 3.20)

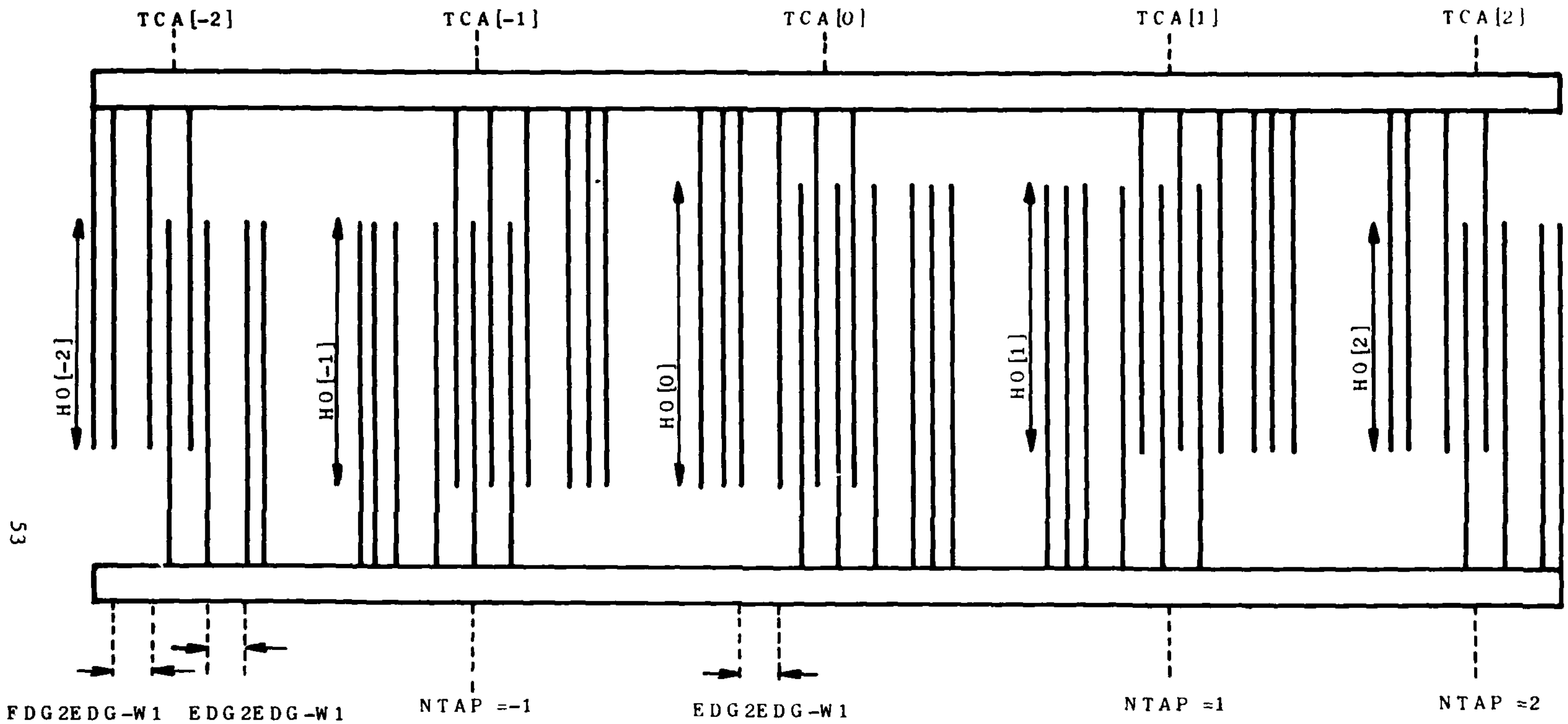


Figure 3.20 Coded Transducer, Lever = True, NF = 6, Single = True, NCHIPS = 4

EDG2EDG(I): This value is used only when CODE(I) = True and CARR(I) = True or when CODE(I) = True and LEVER(I) = True. (CODE(I) and LEVER(I) cannot both be true, program halts if this condition occurs.) See preceding two sections for description.

ADD(I): ADD(I) is used only if CODE(I) = True. The number of elements in the end taps of a coded transducer is a function of NF(I) and NCHIPS(I) as described in Section 3.5.1. ADD(I) is a means of altering the number of elements in the end taps. If ADD(I) = 0 (DEFAULT), then the end taps remain as usual. If ADD(I) = 1, then one element gets added on to the outside of the end tap on the positive side of the comb. If ADD(I) = 2, then one element is added on to the outside of each end tap. In general, if ADD(I) is an even number, then $\frac{ADD(I)}{2}$ elements get added onto each end tap. If ADD(I) is an odd number then $\frac{ADD(I)+1}{2}$ elements get added onto the outside of the end tap on the positive side, and $\frac{ADD(I)-1}{2}$ elements get added onto the outside of the end tap on the negative side.

Neither the position of the so-called "center" of the end tap relative to the edge of the end tap facing the zero'th tap nor the so-called "center' of the end tap relative to the zero'th tap changes. (For definition of "center", see Section 3.5.1.)

That is, if we were to draw a picture of a coded transducer with ADD=0, and then draw the same coded transducer except with ADD=1 (or any other number greater than zero), the second drawing when placed on top of the first drawing would exactly cover the first drawing except for the extensions of the added elements.

The extra elements are added in exactly the manner one would expect. The elements added are always the same type as the tap it is added to. The distance between all elements of the tap including the added elements is W1. (W1 is described in Section 3.5.1). The element lengths and the bus bar they are connected to are such that the overlap between all the elements of the tap is constant (see Figure 3.21).

FILL(I): FILL(I) is used only when CODE(I) = True, LEVER(I), CARR(I) = False. If FILL(I) = True and the above

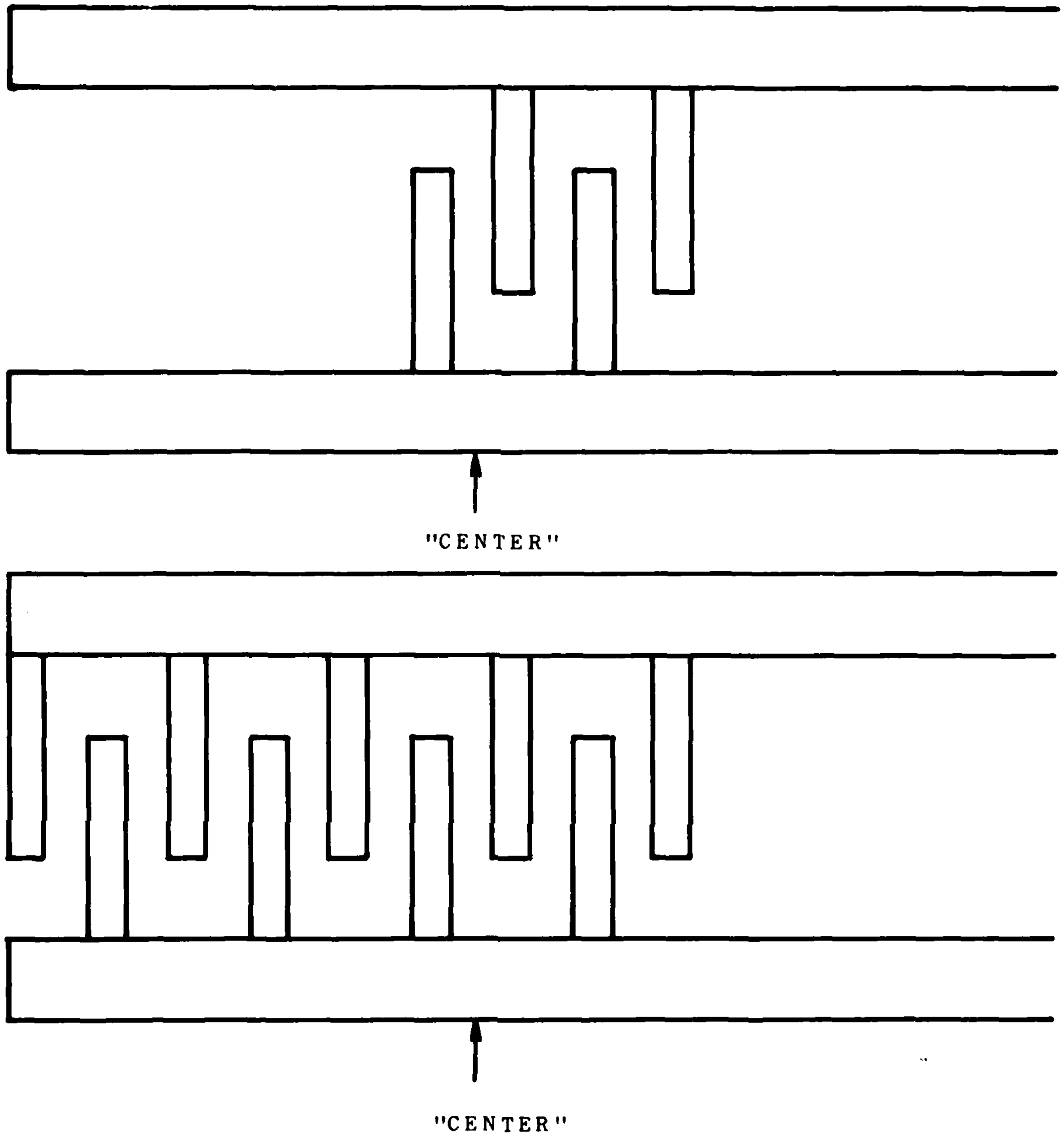


Figure 3.21 Transducer Using Non Zero Value of Variable "ADD"

Original transducer has NF=6 and odd number of taps. Above is a normal end tap with ADD=0. Below the end tap from the same transducer only with ADD=10 (five fingers added). Relative position of "CENTER" from right edge (the edge closes to the zero'th tap) is the same as in the top transducer. In both cases the distance from the "Center" of the end tap to the center of the zero'th tap (TCA[-NTAPMIN]) is the same.

conditions are met, then the spaces between the taps are filled with elements. The rules for filling these spaces are as follows:

A) The type of elements used to fill the spaces (single or double electrodes, with or without dummy fingers) are always the same type of elements used in the rest of the comb.

B) All the FILL elements between two taps are connected to the same bus bar. They are connected to the same bus bar as both the rightmost element of the tap on their left and the leftmost element of the tap on their right.

C) All the FILL elements are the same length. They are the same length as both the rightmost element of the tap on their left, and the leftmost element of the tap on their right.

D) Starting from the rightmost element of the left tap, no matter which side of the comb is negative, the FILL elements are placed at intervals of Distance W1 (W1 is described in Section 3.5.1), until there is no more room.

E) The distance between the rightmost FILL element and the leftmost element of the tap on its right can range in the interval

$$0 \leq \text{DISTANCE} < W1$$

Thus these last two elements may actually touch (see Figure 3.22).

ENGAN(I): When CODE(I) = True, ENGAN(I) = True, and NF(I) is an odd number the Namelist Variable "SINGLE" is ignored and the program automatically alternates between single and double electrodes. The rules describing ENGAN are as follows (see Figure 3.23):

A) If NF is even the program halts and an error message is printed out.

B) The double electrodes are always connected to the bottom bus bar, the single electrodes are always connected to the top bus bar.

C) Both dummy and non-dummy electrodes may be used, but whichever one is used is consistent throughout the whole comb (i.e. non-dummy and dummy electrodes cannot both be used in a comb).

D) If the number of taps in the comb is even then the end taps have the same number of elements as the other taps. (A double electrode in this case still counts as only one element.)

E) If the number of taps in the comb is odd then the end taps have (NF+1)/2 fingers.

F) The distance between any two consecutive fingers is W1.

G) Other rules applying to non-ENGAN taps apply to ENGAN taps (see Section 3.5.1).

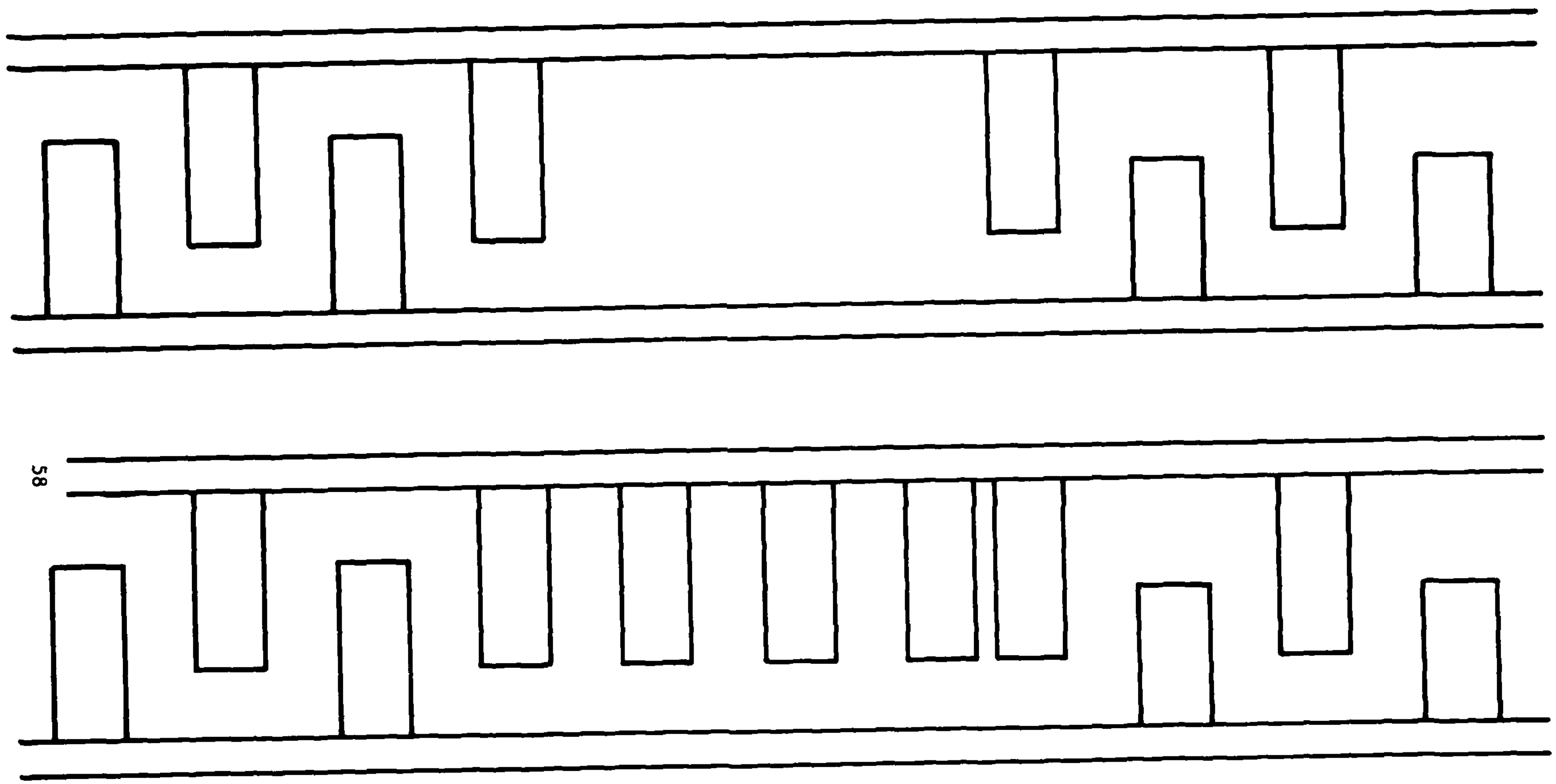


Figure 3.22 Examples of Coded Transducers with FILL=False and FILL=True

The first example (above) is two non-end taps of a coded transducer with FILL=False. The second example is the same two taps of the same transducer, except that FILL=True.

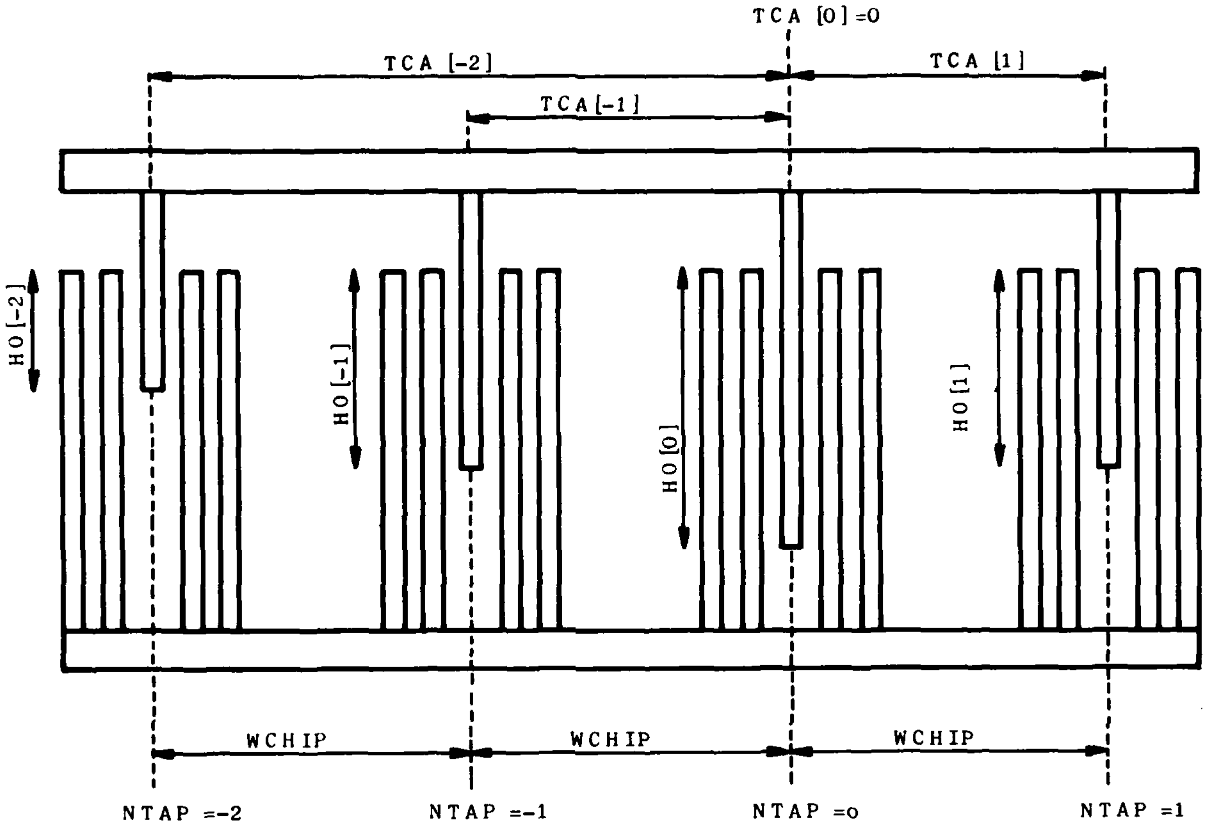


Figure 3.23 A Coded Transducer with ENGAN Type Elements

CODE = True, NCHIPS = 3, NF = 3, ENGAN = True

3.7 Interface with the Raytheon Format

3.7.1. TAPE9

The output of Program CONVERT is a description of the device in the Raytheon format. This information goes on TAPE9 and can either be a physical nine track tape or a temporary disk file.

3.7.2 Coordinate System of the Raytheon Format

The Raytheon format is based on the simple coordinate system shown in Figures 3.24 and 3.25.

The X=0 position is given by the left edge of the leftmost finger. The Y=0 position is given by the top of the bottom pad.

The units used along the X-axis are in meters.

The Y-axis, however, is scaled. The overlap between the M=1/2 and M=-1/2 fingers is assumed to be 1. The Y=1 position is thus given by the top of the M=1/2 finger minus EXTRA, where EXTRA is the distance between the M=1/2 finger and the bottom of the top pad (or the distance between the M=-1/2 finger and the top of the bottom pad). The difference between "EXTRA" and "PADS" is that "PADS" is that distance in real units (usually meters) where as EXTRA is the scaled distance. EXTRA = PADS/OVALAP (where OVALAP is the length in real units of the overlap between the M=-1/2 and M=1/2 fingers. In general, the real dimensions of the comb along the Y-axis are given by multiplying the scaled values times the variable OVALAP.

A possible area of confusion is the fact that the printed output of CONVERT gives a Raytheon description of the comb based on a different coordinate system. For purposes of printed output Y=0 is defined as the bottom of the M=-1/2 element. Everything else is the same (i.e. X=0, X units, and Y scale factor same as described above).

TAPE9 is not the same as the printed output. Do not get confused by thinking they are the same format.

3.7.3 Raytheon Description of Single Fingered Elements

The Raytheon format, for the case with no dummy fingers, simply gives the left X coordinate, the right X coordinate, the bus bar it is connected to, the Y bottom coordinate and the Y top coordinate for each finger in the comb. Thus, if the dimensions of the comb were given in meters, and one graph square represented one millimeter, then the

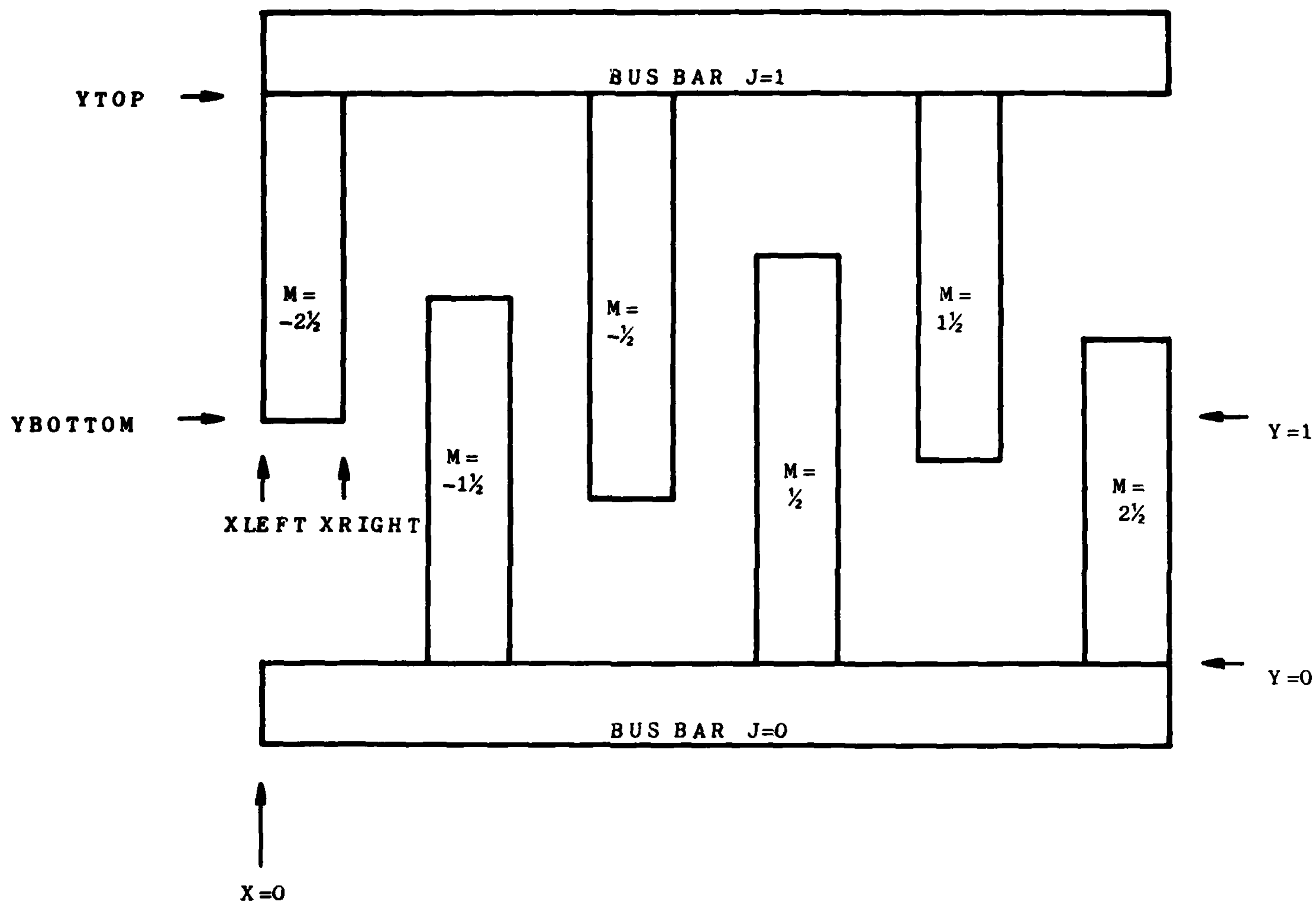


Figure 3.24 <u>Case 1. No Dummy Fingers</u>

X=0 is the left edge of the leftmost finger. Y=0 is given by the top of the botton bus bar. Y-axis is scaled. X-axis uses real units.

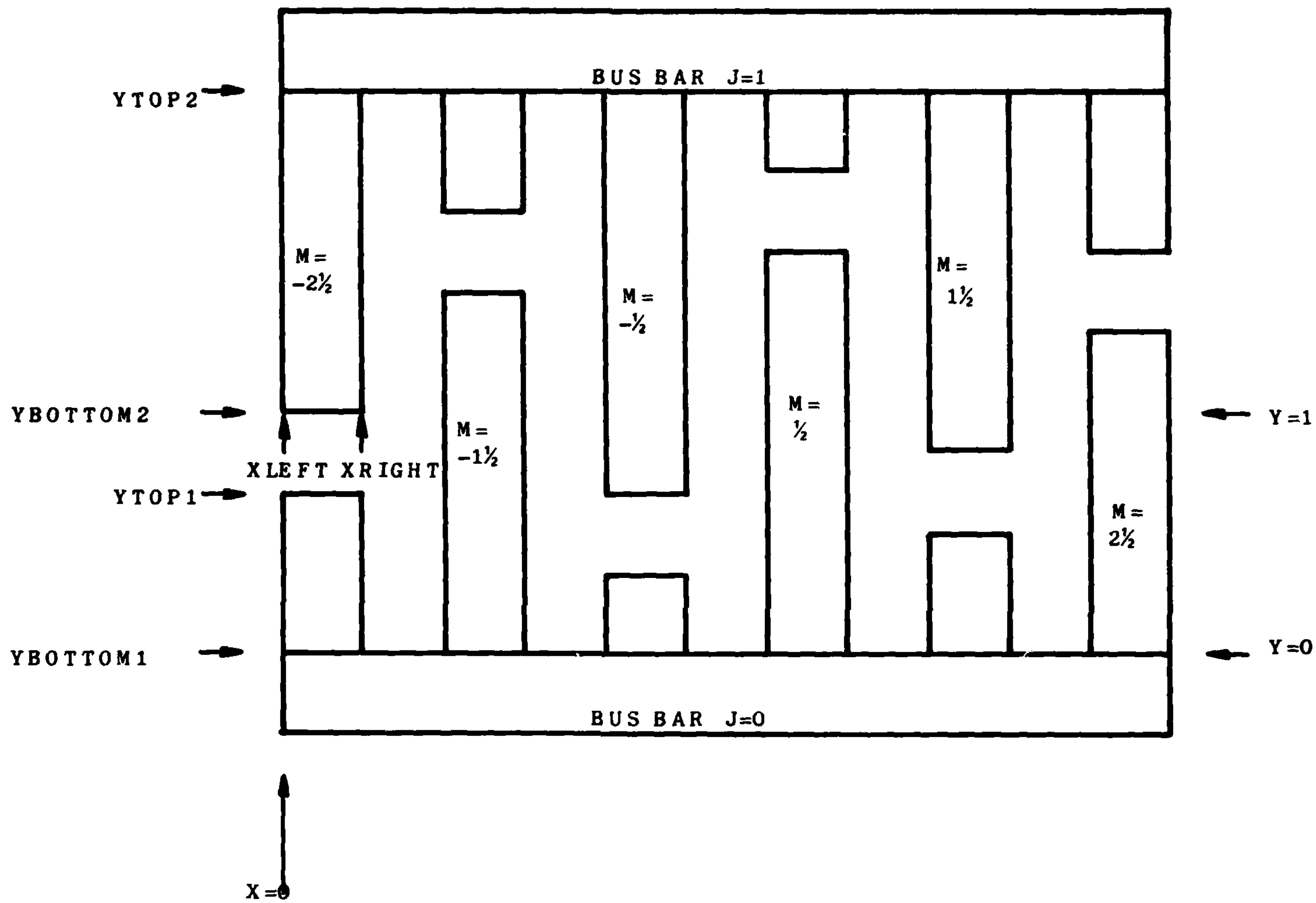


Figure 3.25 Case 2. Dummy Fingers

Raytheon description of the comb in Figure 3.24 would be as follows:

XLeft	XRight	Bus Bar	YBottom	YTOP
0	2E-3	1	1	2.333
4E-3	6E-3	0	0	1.5
8E-3	10E-3	1	.666	2.333
12E-3	14E-3	0	0	1.666
16E-3	18E-3	1	.833	2.333
20E-3	22E-3	0	0	1.333

This line would not actually be written on TAPE9

The actual format of these values on TAPE9 would be:

E15.8, E15.8, I2, E15.8, E15.8, 38(1H)

3.7.4 Raytheon Description of Double Fingered Elements

The Raytheon format, for the case with dummy fingers, gives the left X coordinate, the right X coordinate, the number "zero" meaning the next two coordinates apply to the bottom bus bar, Y coordinate of bottom of finger connected to bottom bus bar, Y coordinate of top of finger connected to the bottom bus bar, the number "one" meaning the next two coordinates apply to the top bus bar, Y coordinate of bottom of finger connected to top bus bar, Y coordinate of top of finger connected to top bus bar.

Thus, again assuming one graph square represents one millimeter, and dimensions of device are in meters, the Raytheon description of the comb in Figure 3.25 would be as follows:

XLeft	XRight	Bus Bar	YBot	YTOP	Bus Bar	YBot	YTop
0	2	0	0	.666	1	1	2.333
4E-3	6E-3	0	0	1.5	1	1.833	2.333
8E-3	10E-3	0	0	.333	1	.666	2.333
12E-3	14E-3	0	0	1.666	1	2	2.333
16E-3	18E-3	0	0	.5	1	.833	2.333
20E-3	22E-3	0	0	1.333	1	1.666	2.333

(This line would not actually be written on TAPE9)

The actual format of these values on TAPE9 would be:

(E15.8, E15.8, I2, E15.8, E15.8, I2, E15.8, E15.8, 6(1H))

Thus, in either case, dummy fingers or no dummy fingers, a 100 character record is written to TAPE9.

In the Raytheon format double fingered elements are handled the same way as single fingered elements. That is, a double electrode is not treated as a single element, but rather as two separate fingers, the coordinates of each being given in the above-described manner. (Note: The printed output of CONVERT distinguishes, for the convenience of the user, between single and double electrodes. Do not get this confused with the Raytheon format on TAPE9.)

3.7.5 More than One Comb per Device

There is usually more than one comb per SAW device. When this is the case the descriptions of the combs on TAPE9 are separated by end of file marks. (There is a maximum of five combs per device.)

3.7.6. Namelist $CONVERT Needed to Complete Raytheon Description

TAPE9 alone is not enough to fully describe a SAW device. For example: How wide are the bus bars? What are the real lengths of the fingers (calculated by OVALAP)?

The $CONVERT namelist is needed to complete the description of the SAW device. The $CONVERT namelist is stored on TAPE28. The write statement used to store the namelist is simply "Write(28,CONVERT)". Thus, TAPE28 can also be read with the simple statement "Read(28,CONVERT)", assuming of course that the $CONVERT namelist is properly defined in the program reading TAPE28.

3.7.7 TAPE29

For use by various programs to be described later (HUGHES, HUGHESCHK), the title cards of the Ho[N] or Ho[NTAP] and ACT[M] or TCA[NTAP] decks are stored on TAPE29. Along with the CONVERT variables EXTRA, DUMMY, SINGLE, FID and REV.

3.8. Conversion Algorithms: RADC Standard to Raytheon

The conversion from the RADC standard format to the Raytheon format is a fairly straightforward process. There are four possible cases.

1) Single electrodes, no dummy fingers.

2) Double electrodes, no dummy fingers.

3) Single electrodes, dummy fingers.

4) Double electrodes, dummy fingers.

Examples of the first two cases will be shown. Algorithms for all four cases will be given.

3.8.1 Conversion of Single Electrodes, No Dummy Fingers (See Figure 3.26)

YB[M] = Y coordinate of bottom of M'th finger

YT[M] = Y coordinate of top of M'th finger

XL[M] = X coordinate of left edge of M'th finger

XR[M] = X coordinate of right edge of M'th finger.

For definitions of W1, W2 and W3, see Section 3.4.3.

The scaled distance from the top of the bottom bus bar to the bottom of the M=-1/2 finger is called EXTRA. It is computed by the equation EXTRA = PADS/OVALAP. Both PADS and OVALAP are on the $CONVERT namelist. EXTRA is also the distance from the bottom of the top bus bar to the top of the M=1/2 finger. In Figure 3.26, EXTRA = 2/3. All fingers touching the bottom bus bar have YB coordinates of zero, thus

$$\begin{aligned} YB(-1.5) &= 0 \\ YB(.5) &= 0 \\ YB(2.5) &= 0 \end{aligned}$$

All fingers touching the top bus bar have YT coordinates of 1 + 2 (EXTRA), thus,

$$\begin{aligned} YT(-2.5) &= 2.333 \\ YT(-.5) &= 2.333 \\ YT(1.5) &= 2.333 \end{aligned}$$

The rest of the Y coordinates can be calculated as follows:

$$\begin{aligned} YB(.5) &= EXTRA = .666 \\ YT(.5) &= YB(-.5) + Ho[0] = 1.666 \end{aligned}$$

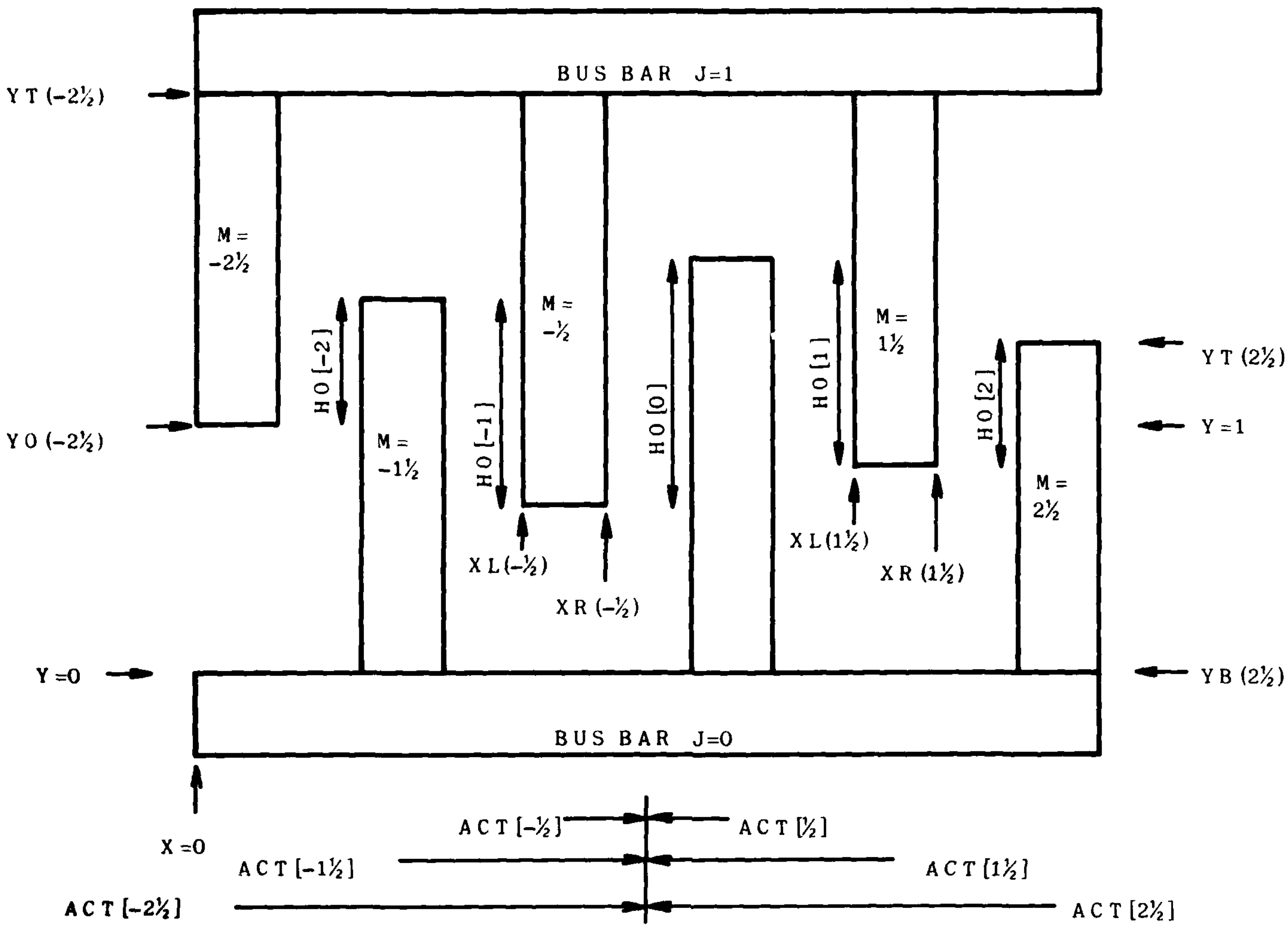


Figure 3.26 Comb with Single Electrodes and no Dummy Elements

Assuming Units of Realistic Distance along X-axis

ACT[-2½]	= -10E-6	$H_o[-2]$	= .5
ACT[-1½]	= -6E-6	$H_o[-1]$	= .833
ACT[-½]	= -2E-6	$H_o[0]$	= 1
ACT[½]	= 2E-6	$H_o[1]$	= .833
ACT[1½]	= 6E-6	$H_o[2]$	= .5
ACT[2½]	= 10E-6		

YB(1.5) = YT(.5) - Ho[1] = .833
YT(2.5) = YB(1.5) + Ho[2] = 1.333
YT(-1.5) = YB(-.5) + Ho[-1] = 1.5
YB(-2.5) = YT(-1.5) - Ho[-2] = 1

To find the XL coordinates we note that the distance "D" from the left edge of the M=-2.5 finger to the center of the zero'th gap is as follows:

$$D = \frac{W1[-2.5]}{2} - ACT[-2.5] = 11E-3$$

Thus for all M; XL[M] = D + ACT[M] - W1[M /2]

Thus	XL(-2.5)	=	0
	XL(-1.5)	=	4E-3
	XL(-.5)	=	8E-3
	XL(.5)	=	12E-3
	XL(1.5)	=	16E-3
	XL(2.5)	=	20E-3

We can now find XR[M] by the equation"

XR[M] = XL[M] + W1[M]

Thus,

XR(-2.5)	=	2E-3
XR(-1.5)	=	6E-3
XR(-.5)	=	10E-3
XR(.5)	=	14E-3
XR(1.5)	=	18E-3
XR(2.5)	=	22E-3

To generalize this case:

A) For all fingers M, touching the bottom bus bar:

YB[M] = 0

B) For all fingers M, touching the top bus bar:

YT[M] = 1 + (2) (EXTRA)

The rest of the Y-coordinates can be figured out inductively (working from the middle out) from the following rules (C thru J).

C) YB(-.5) = EXTRA

D) YT(.5) = 1 + EXTRA

YB(1.5)	=	YT(.5)	-	Ho[1]	=	.833
YT(2.5)	=	YB(1.5)	+	Ho[2]	=	1.333
YT(-1.5)	=	YB(-.5)	+	Ho[-1]	=	1.5
YB(-2.5)	=	YT(-1.5)	-	Ho[-2]	=	1

To find the XL coordinates we note that the distance "D" from the left edge of the M=-2.5 finger to the center of the zero'th gap is as follows:

$$D = \frac{W1[-2.5]}{2} - ACT[-2.5] = 11E-3$$

Thus for all M; XL[M] = D + ACT[M] - W1[M /2]

Thus
XL(-2.5) = 0
XL(-1.5) = 4E-3
XL(-.5) = 8E-3
XL(.5) = 12E-3
XL(1.5) = 16E-3
XL(2.5) = 20E-3

We can now find XR[M] by the equation"

XR[M] = XL[M] + W1[M]

Thus,

XR(-2.5) = 2E-3
XR(-1.5) = 6E-3
XR(-.5) = 10E-3
XR(.5) = 14E-3
XR(1.5) = 18E-3
XR(2.5) = 22E-3

To generalize this case:

A) For all fingers M, touching the bottom bus bar:

YB[M] = 0

B) For all fingers M, touching the top bus bar:

YT[M] = 1 + (2) (EXTRA)

The rest of the Y-coordinates can be figured out inductively (working from the middle out) from the following rules (C thru J).

C) YB(-.5) = EXTRA

D) YT(.5) = 1 + EXTRA

E) If M is positive and finger M is connected to the same bus bar as finger (M-1) then

$$YT[M] = YT[M-1]$$
$$YB[M] = YB[M-1]$$

F) If M is positive and finger M is connected to the top bus bar and finger (M-1) is connected to the bottom bus bar

$$YB[M] = YT[M-1] - Ho[M-1/2]$$

G) If M is positive and finger M is connected to the bottom bus bar and finger (M-1) is connected to the top bus bar

$$YT[M] = YB[M-1] + Ho[M-1/2]$$

H) If M is negative and finger M is connected to the same bus bar as finger (M+1) then

$$YT[M] = YT[M+1]$$
$$YB[M] = YB[M+1]$$

I) If M is negative and finger M is connected to the top bus bar and finger (M+1) is connected to the bottom bus bar

$$YB[M] = YT[M+1] - Ho[M+1/2]$$

J) If M is negative and finger M is connected to the bottom bus bar and finger (M+1) is connected to the top bus bar

$$YT[M] = YB[M+1] + Ho[M+1/2]$$

K) Let $D = \frac{W1[-NMIN-1/2]}{2} - ACT[-NMIN-1/2]$

(-NMIN-1/2 is equal to the minimum M value (see Section 3.4.2).

L) $XL[M] = D + ACT[M] - W1[M /2]$ For all M

M) $XR[M] = XL[M] + W1[M]$ For all M.

Note: ACT[M] values may be explicitly defined by an ACT[M] deck or implicitly defined as described in Section 3.4.5.

3.8.2 <u>Conversion of Double Electrodes, No Dummy Fingers</u> (See figure 3.27)

XL1[M] = X coordinate of left edge of left finger of M'th double electrode.

XL2[M] = X coordinate of left edge of right finger of M'th double electrode.

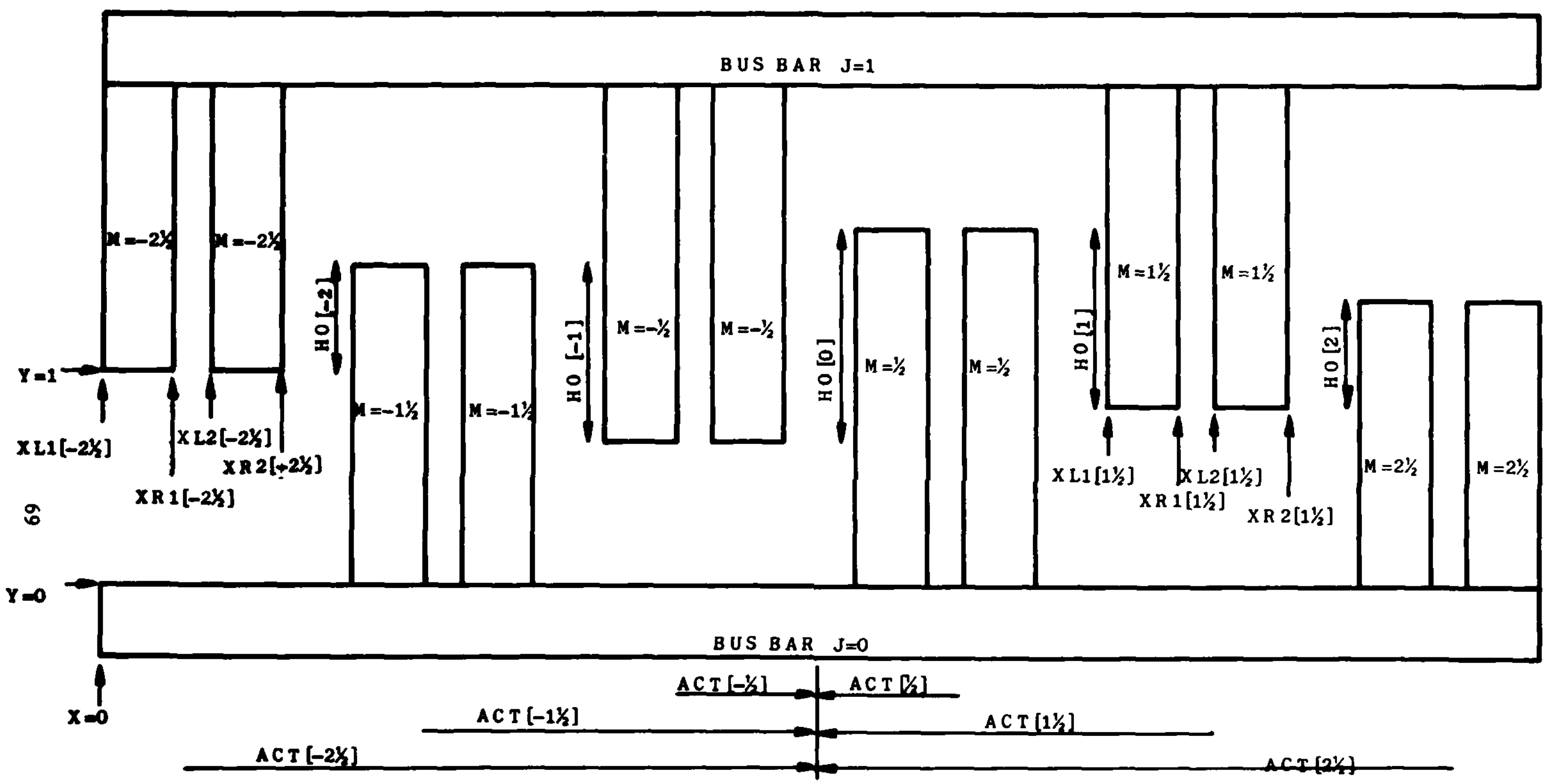


Figure 3.27 <u>Comb with Double Electrodes and No Dummy Fingers</u>

Assuming Units of Realistic Distance along X-axis

ACT[-2½]	= -17.5E-6	$H_o[-2]$	= .5
ACT[-1½]	= -10.5E-6	$H_o[-1]$	= .833
ACT[-½]	= -3.5E-6	$H_o[0]$	= 1
ACT[½]	= 3.5E-6	$H_o[1]$	= .833
ACT[1½]	= 10.5E-6	$H_o[2]$	= .5
ACT[2½]	= 17.5E-6		

XR1[M] = X coordinate of right edge of left finger of M'th double electrode.

XR2[M] = X coordinate of right edge of right finger of M'th double electrode.

To find the YB[M] and YT[M]values we use the exact same algorithm as described in Section 3.8.1.

To find the XL1, XL2, XR1 and XR2 coordinates we note that the Distance "D" from the left ede of the M = -2.5 double electrode to the center of the zero'th gap is as follows:

$$D = W2(-2.5) + \frac{W1\ -2.5}{2} - ACT(-2.5).. = 20E-3$$

Thus for all M

$$XL1[M] = D + ACT[M] - W2[M] - \frac{W1[M]}{2}$$

$$XL2[M] = D + ACT[M] + \frac{W1[M]}{2}$$

Thus,

XL1(-2.5)	= 0	XL2(-2.5)	= 3E-3
XL1(-1.5)	= 7E-3	XL2(-1.5)	= 10E-3
XL1(-.5)	= 14E-3	XL2(-.5)	= 17E-3
XL1(.5)	= 21E-3	XL2(.5)	= 24E-3
XL1(1.5)	= 28E-3	XL2(1.5)	= 31E-3
XL1(2.5)	= 35E-3	XL2(2.5)	= 38E-3

We can now find XR1[M] and XR2[M] by the following equations.

$$XR1[M] = XL1[M] + W2[M]$$

$$XR2[M] = XL2[M] + W2[M]$$

Thus,

XR1(-2.5)	= 2E-3	XR2(-2.5)	= 5E-3
XR1(-1.5)	= 9E-3	XR2(-1.5)	= 12E-3
XR1(-.5)	= 16E-3	XR2(-.5)	= 19E-3

XR1(.5)	=	23E-3	XR2(.5)	=	26E-3
XR1(1.5)	=	30E-3	XR2(1.5)	=	33E-3
XR1(2.5)	=	37E-3	XR2(2.5)	=	40E-3

To generalize,

A) Let $D = W2[-NMIN-1/2] + \frac{W1[-NMIN-1/2]}{2} - ACT[-NMIN-1/2]$

(-NMIN-1/2 is equal to minimum M value (see Section 3.4.2).)

B) $XL1[M] = D + ACT[M] - W2[M] - \frac{W1[M]}{2}$ for all M

C) $XL2[M] = D + ACT[M] + \frac{W1[M]}{2}$ for all M

D) $XR1[M] = XL1[M] + W2[M]$ for all M

E) $XR2[M] = XL2[M] + W2[M]$ for all M

3.8.3 Conversion of Single and Double Electrodes that Use Dummy Fingers (See Figures 3.3, 3.4, 3.26 and 3.27)

All the coordinates of the main (electrically active) elements can be calculated using the exact same algorithms described in Sections 3.8.1 and 3.8.2. The coordinates of the dummy elements can be easily calculated using the fact that the scaled distance between the M'th main and dummy fingers, called W3 gap, is equal to W3[M]/OVALAP.

The X coordinates of the dummy finger will be the same as the X coordinates of the corresponding main finger. To find the YTop coordinate of a dummy finger connected to the bottom bus bar, simply subtract W3 gap from the YBottom coordinate of the corresponding main finger. To find the Ybottom coordinate of a dummy finger connected to the top bus bar simply add W3 gap to the YTop coordinate of the corresponding main finger.

3.9 Conversion Algorithms: RADC Coded to Raytheon

There are many different combinations of options that can be used with coded transducers. In each parenthesis below one of the options may be chosen.

$$\begin{pmatrix}\text{SINGLE=T}\\ \text{SINGLE=F}\\ \text{ENGAN=T}\end{pmatrix}\begin{pmatrix}\text{DUMMY=T}\\ \\ \text{DUMMY=F}\end{pmatrix}\begin{pmatrix}\text{NTAPS is Odd}\\ \\ \text{NTAPS is Even}\end{pmatrix}\begin{pmatrix}\text{NF is Odd}\\ \\ \text{NF is Even}\end{pmatrix}\begin{pmatrix}\text{CARR,LEVER,FILL=False}\\ \text{CARR=True}\\ \text{LEVER=True}\\ \text{FILL=True}\end{pmatrix}\begin{pmatrix}\text{ADD=0}\\ \\ \text{ADD}\neq 0\end{pmatrix}$$

$3 \times 2 \times 2 \times 2 \times 4 \times 2 = 192$ combinations.

Actually $\begin{pmatrix}\text{SINGLE=T}\\ \text{SINGLE=F}\\ \text{ENGAN=T}\end{pmatrix}\begin{pmatrix}\text{NF Odd}\\ \\ \text{NF Even}\end{pmatrix}$ only represents five possibilities because ENGAN=T, NF=Even is illegal; thus, there are 160 combinations.

Obviously all 160 algorithms cannot be given. The generalized algorithm for handling all these cases is quite lengthy. In the following section (Section 3.9.1), one example conversion will be given.

In Section 3.9.2, some generalizing principals will be given. It is hoped that by studying Sectons 3.5, 3.8, 3.9.1 and 3.9.2, the reader will be able to understand the conversion process for any of the 160 cases.

3.9.1 Example of Conversion of Coded Transducer with Single Electrodes and No Dummy Fingers (See Figure 3.28)

NTAP = The number of the tap.

M = The number of the finger of the tap (counting from the left).

YB(NTAP,M) = The Y coordinate of the bottom of the M'th finger of the NTAP'th tap.

YT(NTAP,M) = The Y coordinate of the top of the M'th finger of the NTAP'th tap.

XL(NTAP,M) = The X coordinate of the left edge of the M'th finger of the NTAP'th tap.

XR(NTAP,M) = The X coordinate of the right edge of the M'th finger of the NTAP'th tap.

For a definition of EXTRA see Sections 3.6.4 (under PADS) and 3.8.1.

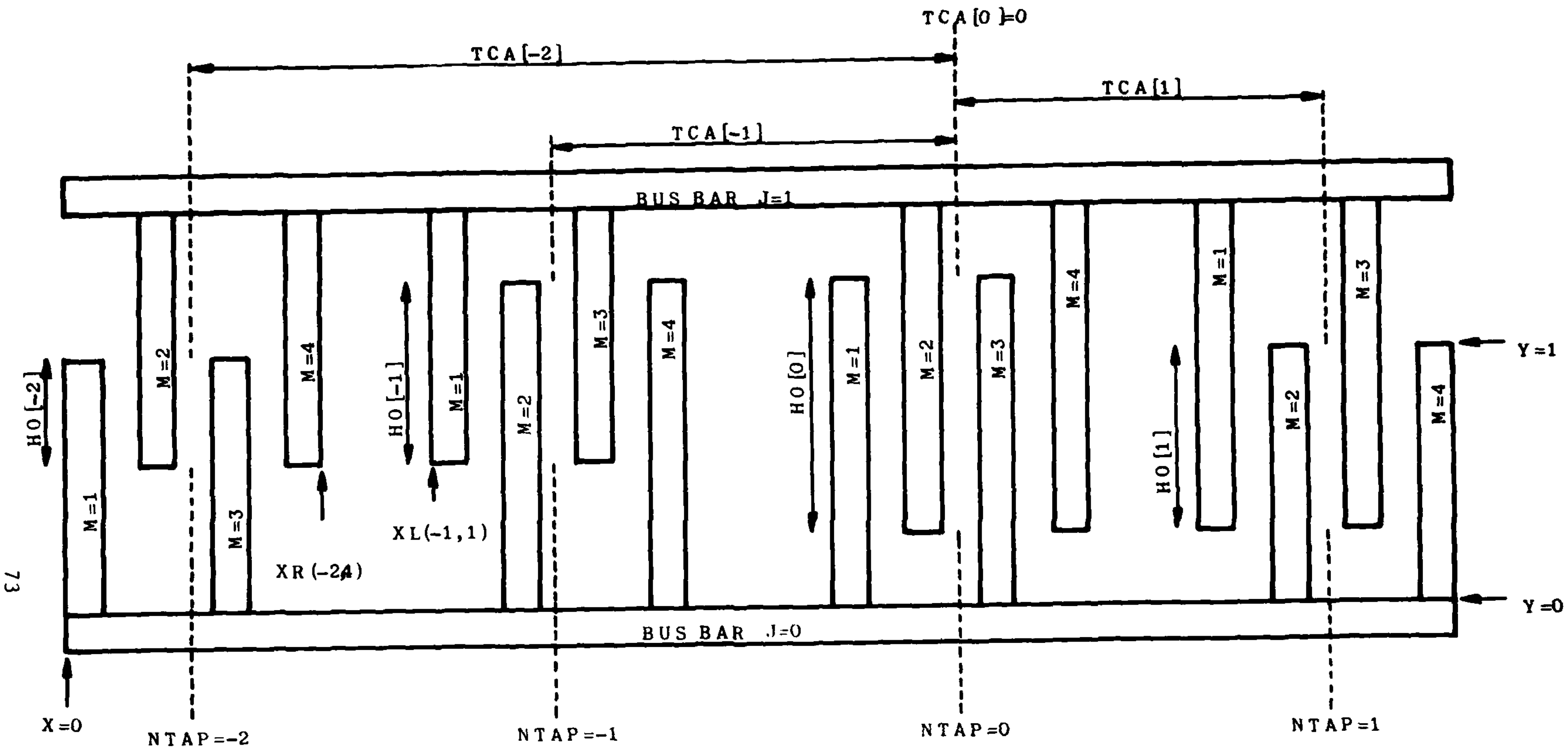


Figure 3.28 Coded Transducer, NTAPS=4, NF=4

Assuming Units of Realistic Distance along X-axis, then,

TCA[-2]	=	-21E-6	$H_o[-2]$	=	.428
TCA[-1]	=	-11E-6	$H_o[-1]$	=	.714
TCA[0]	=	0	$H_o[0]$	=	1
TCA[1]	=	10E-6	$H_o[1]$	=	.714

In Figure 3.28, EXTRA = .286

All fingers touching the bottom bus bar have YB coordinates of zero, thus,

YB(-2,1) = 0	YB(0,1) = 0
YB(-2,3) = 0	YB(0,3) = 0
YB(-1,2) = 0	YB(1,2) = 0
YB(-1,4) = 0	YB(1,4) = 0

All fingers touching the top bus bar have YT coordinates of 1 + (2) (EXTRA), thus

YT(-2,2) = 1.571	YT(0,2) = 1.571
YT(-2,4) = 1.571	YT(0,4) = 1.571
YT(-1,1) = 1.571	YT(1,1) = 1.571
YT(-1,3) = 1.571	YT(1,3) = 1.571

The rest of the Y-coordinates can be calculated as follows:

By definition of EXTRA: YB(0,2) = EXTRA = .286
YB(0,4) = EXTRA = .286

The overlap in the zero'th tap is always 1, thus

YT(0,1) = 1 + EXTRA = 1.286
YT(0,3) = 1 + EXTRA = 1.286

The two elements facing each other of two consecutive taps are always the same length (see Section 3.5.1).

Thus

YB(1,1) = YB(0,4) = .286
YB(1,3) = YB(1,1) = .286
YT(1,2) = YB(1,1) + Ho[1] = .286 + .714 = 1
YT(1,4) = YT(1,2) = 1

The two elements facing each other of two consecutive taps are of equal length, thus

YT(-1,4) = YT(0,1) = 1.286
YT(-1,2) = YT(-1,4) = 1.286
YB(-1,1) = YT(-1,2) - Ho[-1] = 1.286 - .714 = .571
YB(-1,3) = YB(-1,1) = .571

Facing elements of two consecutive taps have the same length, thus

YB(-2,4) = YB(-1,1) = .571
YB(-2,2) = YB(-2,4) = .571
YT(-2,1) = YB(-2,2) + Ho[-2] = .571 + .428 = 1
YT(-2,3) = YT(-2,1) = 1

To find the XR coordinates we note that

XR(NTAP,M) = XL(NTAP,M) + 1 for all NTAP and M

3.9.2 Some General Principals for Conversion of Coded Transducers

The Y-Bottom and Y-Top coordinates can be found by first finding these coordinates for the zero'th tap, and then using the Ho[NTAP] values plus the fact, that facing elements of two consecutive elements are equal, to find all the other Y-Bottom and Y-Top coordinates.

To find the XL coordinates, first find the distance "D" between the left edge of the comb (X = 0 at this point) and the center of the zero'th tap.

For taps with ADD=0, FILL=F, CXARR=F, and LEVER=F, "D" can be calculated as follows:

$D = -TCA[-NTAPMIN] + \dfrac{2(NF)-1}{2}(W1)$	NF = Even or Odd Even number of taps Single Electrodes
$D = -TCA[-NTAPMIN] + (11/2)(W1)$	NF = Even Odd Number of Taps Single Electrodes
$D = -TCA[-NTAPMIN] + (1/2)(W1)$	NF = Odd Odd Number of Taps Single Electrodes
$D = -TCA[-NTAPMIN] + (NF)(W2) + \dfrac{2(NF)-1}{2}(W1)$	NF = Even or Odd Even Number of Taps Double Electrodes
$D = -TCA[-NTAPMIN] + (2)(W2) + (1\ 1/2)(W1)$	NF = Even Odd Number of Taps Double Electrodes
$D = -TCA[-NTAPMIN] + W2 + (1/2)(W1)$	NF = Odd Odd Number of Taps Double Electrodes

Note: NF = Number of Elements per tap, not number of fingers.

We can now calculate DTAP[NTAP]

DTAP[NTAP] = D + TCA[NTAP]

The left and right coordinates (XL and XR) of any finger in the NTAP'th tap can now be found by adding or subtracting the appropriate multiples of

$$\frac{W1}{2} \text{ and } \frac{W2}{2} \quad \text{to DTAP[NTAP]}$$

For taps with various options such as CARR, LEVER, FILL, ENGAN or ADD the coordinates of the added fingers are found by applying the rules of these options to the values of W1, W2 and EDG2EDG.

3.10 Example Input Setups for Program CONVERT

For each comb there are seven possible input formats (see Section 3.2). They are:

A)	Ho[N] deck followed by ACT[M] deck	(RADC Standard Format)
B)	Ho[N] deck alone	(RADC Standard Format)
C)	ACT[M] deck alone	(RADC Standard Format)
D)	Ho[NTAP] deck followed by TCA[NTAP] deck	(RADC Coded Format)
E)	Ho[NTAP] deck alone	(RADC Coded Format)
F)	TCA[NTAP] deck alone	(RADC Coded Format)
G)	The Null Set (no decks)	(RADC Coded Format)

The first card of a CONVERT input setup may either be a title card containing any message up to 80 characters long (to appear in output listings and plots) or the $CONVERT namelist. The title card is optional.

Title cards for the Ho[N], Ho[NTAP], ACT[M] and TCA[NTAP] decks are not optional. They are a mandatory part of the above decks.

3.10.1 Three combs, All Format A

Optional Title Card
$CONVERT Namelist
Ho[N] Deck
789 Card
ACT[M] Deck
789 Card
Ho[N] Deck
789 Card
ACT[M] Deck
789 Card
Ho[N] Deck
789 Card
ACT[M] Deck
789 Card

3.10.2 Two Combs, Formats B and C

Optional Title Card
$CONVERT Namelist
Ho[N] Deck
789 Card
789 Card
789 Card
ACT[M] Deck
789 Card

3.10.3 Four Combs, Formats D, E, F, G

Optional Title Card
$CONVERT Namelist
Ho[NTAP] Deck
789 Card
TCA[NTAP] Deck
789 Card
789 Card
TCA[NTAP] Deck
789 Card
Ho[NTAP] Deck
789 Card
789 Card
789 Card
789 Card

3.10.4 <u>Four Combs, Formats A, G, G, G</u>

Optional Title Card
$CONVERT Namelist
Ho[N] Deck
789 Card
ACT[M] Deck
789 Card
789 Card
789 Card
789 Card
789 Card
789 Card
789 Card

3.10.5 General Rules

There are always twice as many 789 cards as the number of combs. A 789 card follows every Ho[N], ACT[M], Ho[NTAP] and TCA[NTAP] deck. If one of these decks is left out the 789 card is still included. One can think of it in the following way.

For each comb there is either an Ho[N] or Ho[NTAP]deck followed by either an ACT[M] or TCA[NTAP]deck. Every deck ends with a 789 card. Sometimes this 789 card is the only card in the deck.

4.0 PROGRAM CONVXY

4.1 Introduction

Program CONVXY is basically a more versatile version of program CONVERT. Program CONVERT is designed to work in conjunction with programs ELECTRO and HUGHES (among others). Program CONVXY is designed to work in conjunction with programs ELECTROXY and HUGHESXY only. Most of what applies to program CONVERT (see Chapter 3.0) also applies to program CONVXY, thus only the differences between these two programs will be covered in this section. Except for the differences described in this section, the description in Section 3.0 applies to CONVXY as well as to CONVERT.

4.2 CONVXY vs CONVERT: Differences from a User's Point of View

4.2.1 More COMBS Allowed Per Device

Program CONVERT allows a maximum of five combs per device ($1 \leq$ NCOMBS ≤ 5). Program CONVXY allows a maximum of sixteen combs per device ($1 \leq$ NCOMBS ≤ 16). As in program CONVERT there must be a set of cards for each comb (see Section 3.2). Default Value: NCOMBS = 2.

4.2.2 Variables OVALAP, PADS May Vary From Comb to Comb

In program CONVERT the variables PADS and OVALAP from the $CONVERT namelist are not arrays and do not vary from comb to comb. In other words in program CONVERT all combs have the same vertical distance between pads (OVALAP + 2*PADS). In program CONVXY these variables are arrays in which the width of each comb may vary the vertical distance between pads of the I'th comb being (OVALAP(I) + 2*PADS(I)).

Default Values:

OVALAP(I) = 500.E-6
PADS (I) = 10.E-6 For I = 1 to 16

4.2.3 *REVERSE(16) is a New $CONVERT Namelist Variable*

In program CONVERT some of the combs are automatically reversed according to the rules described in Section 3.4.6.3. In program CONVXY the user decides which, if any, of the combs are to be reversed.

If REVERSE(I) = .False. then the I'th comb is not reversed.
If REVERSE(I) = .True. then the I'th comb is reversed.

REVERSE is a logical variable.
DEFAULT: REVERSE = F,F,F,F,F,F,F,F,F,F,F,F,F,F,F,F

4.2.4 NMIND(16) and NMAXD(16) are New $CONVERT Namelist Variables

The variables NMIND(16) and NMAXD(16) allow the user to fabricate an RADC standard format type comb with neither an Ho[N] deck nor an ACT[M] deck. Rules for using these variables follow:

A) If the user does not intend to use this option, just ignore these variables. The program will supply default values of 500 which will be ignored. The actual number of fingers will be determined from either the Ho[N] deck or the ACT[M] deck, or both, just as it does in program CONVERT (see Section 3.4.4). Note: If either an ACT[M] deck or an Ho[N] deck is used, then NMIND and NMAXD must equal 500 (the default value).

B) If a comb is being fabricated with the RADC coded format, ignore these variables (Note: The Default values of 500 must be used).

C) If the user wants the I'th comb to be fabricated with neither an Ho[N] nor an ACT [M] deck then set

NMIND[I] = The number of gaps before the zero'th gap. A positive number

NMAXD[I] = The number of gaps after the zero'th gap. A positive number

Once the number of fingers (or gaps) have been calculated the rest of the transducer specifications are calculated according to the rules in Section 3.4.4.

D) If the user wants to devise a multi-comb device, some combs being described by actual physical decks, others using the method described above, then the values of NMIND and NMAXD for the combs made from physical decks should be 500 (their default values).

Example: (See Figure 4.1). One wants to fabricate a five comb device where the second and fourth combs are fabricated without physical decks, but using the RADC standard format. Assume the second and fourth combs are symmetrical around the zero'th gap and both have ten fingers. The first, third and fifth combs are fabricated from physical decks, thus

NMIND = 500, 4, 500, 4, 500
NMAXD = 500, 4, 500, 4, 500

The deck setup might look as follows:
(Combs 1, 3 and 5 could have any of the formats described in Section 3.2.)

Job Card

Control Cards

789

Convert Title Card

$CONVERT Namelist   NMAXD = 500, 4, 500, 4, 500,   NCOMB = 5
NMIND = 500, 4, 500, 4, 500
(and any other namelist options the user includes)

Ho [N] Title Card

Ho [N] Deck

789

ACT [M] Title Card

ACT [M] Deck

789

COMB 1

789

789

COMB 2

Ho[N] Title Card

Ho[N] Deck

COMB 3

789

789

789

COMB 4

789

789

ACT[M] Title Card

COMB 5

ACT[M] Deck

789

NOTE: A comb may also be described using no physical decks by using the RADC coded format (see Sections 3.2.6, 3.5.5).

If the I'th comb is described using a coded format, (i.e. CODE(I) = True) then let

NMIND(I) = 500
NMAXD(I) = 500 (defaults)

for any form of the RADC coded format you use.

4.3 CONVXY vs. CONVERT Differences from a Programmer's Point of View

Programs CONVERT and CONVXY are both almost always run in conjunction with at least one other program. They are seldom, if ever, run by themselves. Programs CONVXY and CONVERT communicate to other programs through three files. These files are the only means of communication.

4.3.1 TAPE9

TAPE9 can be either a temporary disk file or an actual physical nine track tape (see Sections 3.7.1, 3.7.5). The format of the data on TAPE9 is the same for both CONVERT and CONVXY. The only difference is that with CONVERT, TAPE9 has a maximum of five sets of data (one set for each comb), each set being separated by an end of file mark. With CONVXY, TAPE9 has a maximum of sixteen sets of data, each separated by and end of file mark.

4.3.2 TAPE28

Usually a temporary disk file (see Section 3.7.6). The variables OVALAP and PADS (on the $CONVERT namelist) have been changed to arrays. The variables NMIND, NMAXD and REVERSE have been added to the $CONVERT namelist. Thus, if the programmer wishes to use the same method described in Section 3.7.6 to read and write to TAPE28, one must make sure that the $CONVERT namelist is consistently defined in all the interfacing programs.

4.3.3 TAPE29

TAPE29 is usually a temporary disk file (see Section 3.7.7). The format of the data on TAPE29 is the same for both CONVERT and CONVXY. The only difference is that with CONVERT, TAPE29 has a maximum of five sets of data (not separated by end of file marks). With CONVXY TAPE29 has a maximum of sixteen sets of data (not separated by end of file marks).

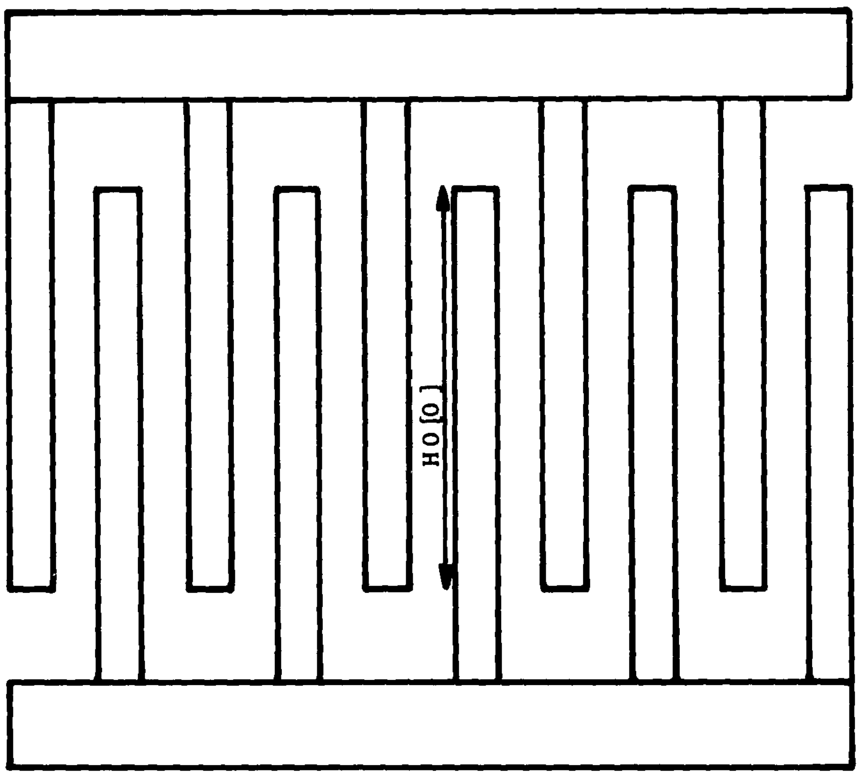


Figure 4.1 Single Electrode, No Dummy Comb

Made as described in Section 4.2.4.D

5.0 SPECIFYING A SAW DEVICE MASTER (STEP AND REPEAT MACHINE)

5.1 Introduction

The programs described in this chapter were originally designed to interface with equipment and procedures at Hughes Aircraft Company. However, these programs should be able to be used, with slight modification, with any Electromask Step and Repeat Machine. The purpose of these programs is to produce printed output, plotted output, and magnetic tapes that allow the Hughes Company to produce a glass "Master", and to allow both the Hughes Company and the device engineers to verify the correctness of the finished master.

A "Master" is a photographic piece of glass with a picture of the pattern that the device engineer eventually wants in the form of a metal pattern on a piezoelectric crystal. For the negative master case, where the glass is clear there will be metal in the final pattern and where the glass is opaque there will be no metal. The size of the pattern on the master is not necessarily the same size as the metal pattern on the finished piezoelectric crystal. Often the pattern on the glass master is magnified by, for example, ten times the final size. A photographic technique is then used to transform the pattern on the master to a physical metal pattern, of the correct size, on a piezoelectric crystal.

The steps in producing a finished Surface Acoustic Wave (SAW) device are as follows:

A) The device engineer describes any device using one of the RADC formats (see Section 3.0, program CONVERT).

B) This format is used as input to program CONVERT, which produces as output, a description of the SAW Device in the Raytheon Format.

C) The Raytheon Format, along with some other device specifications, is used as input to program HUGHES. Program HUGHES produces printed output and a magnetic tape. This magnetic tape contains the instructions that will eventually be used by the Hughes Step and Repeat Machine to produce a glass master.

D) The magnetic tape is used as input to program HUESCHK. HUESCHK is a simulation program that simulates the Step and Repeat Machine at Hughes. As output it produces a plotted picture of the master.

E) The device engineer looks at the plotted picture of the master to make sure it is what one expects. If the device has been incorrectly described, or the magnetic tape is faulty, the problems will hopefully be detected at this point.

F) The magnetic tape is run on an IBM computer at MITRE to eliminate an incompatibility between the tape formats used on the CDC 6600 computing system at Hanscom Air Force Base and the computers used at the Hughes Company.

G) The magnetic tape, the printed output from program HUGHES, and the plotted picture from program HUESCHK, are sent to the Hughes Company. Here, using a duplicated version of the magnetic tape and some parameters from the printed output as input to the Step and Repeat machine, they produce a glass master.

H) The Hughes Company compares the glass master to the plotted picture and to information from the printed output. If they are satisfied that no problems exist with the glass master, they send it to the device engineers (RADC).

I) The device engineers verify the correctness of the glass master.

J) A direct optical projection technique is used to produce the final SAW device as a metal pattern on a piezoelectric crystal.

Program HUGHESXY and program HUESCHKTEK are the other two programs described in this section. Program HUGHESXY is a more versatile version of program HUGHES. Program HUESCHKTEK is similar to program HUESCHK except it produces a picture on the Tektronix machine instead of the CALCOMP plotter.

5.2 Program HUGHES

5.2.1 Introduction

Program HUGHES accepts as input the Raytheon Format from program CONVERT, and the $HUGHES namelist. The program produces printed output to be used by the operators at the Hughes Company and a magnetic tape that gives instructions to the Hughes Step and Repeat Machine. The printed output and the magnetic tape are all the Hughes Company needs to produce the glass master (as described in Section 5.1).

5.2.2 The Hughes Step and Repeat Machine (manufactured by electromask)

The Hughes Step and Repeat Machine functions as follows:

A blank photographic glass master is locked into place on the Step and Repeat Machine. As part of the machine there is a replaceable metal plate with a hole in it. The hole is known as the reticle. The size, shape, and placement of the reticle on the metal plate is specified by the user. For the purposes of program HUGHES, however, the shape is always rectangular and placed in the center of the metal plate. The metal plate is parallel to and directly above the glass master. When the center of the reticle is above the lower left corner of the glass master the coordinate position of the metal plate is defined to be (0,0).

According to instructions from a magnetic tape, the machine may reposition the reticle anywhere above the glass master. The magnetic tape may also tell the machine to "Flash". That is, a light comes on and shines through the reticle exposing the master directly below the reticle. The exposed area will be the exact reduced shape and size of the reticle. Eventually the glass master will be chemically treated and all "flashed" areas become clear. Another instruction from the magnetic tape may cause the machine to pause so that the operator may change the reticle size (by changing the metal plate). When detailed lines or shapes are required, a small reticle is used. When large areas of the master are to be erased a large reticle is used. The different reticles must be requested and "cut out" in advance. The magnetic tape only tells the machine when to pause. The operator must have separate information (in the form of printed computer output for example) that tells him what reticle to use. The information might look as follows: (hypothetically)

Start:	Use $.3 \times 10^{-4} \times .1 \times 10^{-2}$ reticle
Pause 1:	Use $.3 \times 10^{-4} \times .3 \times 10^{-4}$ reticle
Pause 2:	Use $.2 \times 10^{-2} \times .2 \times 10^{-2}$ reticle

5.2.3 Magnetic Tape Format

The tape format necessary to run the Step and Repeat Machine is as follows:

A) The tape is 9 track, 800 bits per inch, odd parity written in fixed records of 798 bytes. The tape is coded in EBCDIC.

B) The tape must be a "No Label" tape. That is, there must be no introductory material such as title of job, coding instructions, etc. The tape must start immediately with the instruction format described.

C) The first character of each record is the single quote (').

D) The last non-blank character in each record is a period (.). Each record has exactly one period. If the period is not the 798'th character of the record, then the remaining characters of the record must be blank.

E) The character "X", followed by from one to seven digits, specifies the X position of the center of the metal plate (containing the reticle). The character "Y", followed by from one to seven digits, specifies the Y position of the center of the plate. These digit sequences may represent either units of laser counts or 10^{-7} meters (an operator option, that must be consistent with the software used to produce the magnetic tape). Program HUGHES is capable of using either set of units; however, laser counts have been adopted as standard. There are 321,103 laser counts per inch. Thus, the command X765432, for example, would place the reticle about 2.38 inches to the right of the origin (i.e., a horizontal distance of 2.38 inches).

The X or Y character sequences described above must never cross record boundaries. Also, they may never end as the 798'th character of a record since the last non-blank character of a record must be a period.

F) The character "s" (followed by no digits) tells the machine to "Flash". Flashing is described in Section 5.2.2.

G) The character "Z" (followed by no digits) tells the machine to pause. The Z command is used to both end the job (the master being complete) or to allow the operator to change reticles. The operator must manually restart the machine following a reticle change. Not all records contain a Z command; however no record ever contains more than one Z command. The Z command, when it is given, must always be the second to last non-blank character in a record (the last non-blank character being a period). Thus, each new reticle must start a new record.

The hardware used on the CDC 6600 computer at Hanscom AFB does not allow tapes to have 798 characters per record. Instead tapes with 800 characters per record are used. The magnetic tapes created contain 798 valid characters per record followed by two blank (filler) characters. The tape is now taken to MITRE where hardware is available to handle both tapes with 798 characters per record and tapes with 800 characters per record. Here the CDC generated tape (with 800 characters per record) is copied onto a new tape (with 798 characters per record) discarding the last two characters of all the 800 character records. It is this new tape that is actually sent to Hughes. (See Figure 5.1 for an example of several records written on a magnetic tape).

5.2.4 Input to Program HUGHES

Input to program HUGHES consists of TAPE9, TAPE28, TAPE29, and the $HUGHES namelist.

5.2.4.1 TAPE9

TAPE9 is usually a temporary disk file (created by program CONVERT). TAPE9 contains a Raytheon description of the SAW device to be fabricated. The format of TAPE9 is discussed in Section 3.7.

5.2.4.2 TAPE28

TAPE28 is usually a temporary disk file (created by program CONVERT). TAPE28 contains the values of the $CONVERT namelist. Many of the variables on the $CONVERT namelist are also needed by program HUGHES. Rather than have the user input these values again, they are passed to program HUGHES via TAPE28. The $CONVERT namelist is described in Section 3.6. TAPE28 is also discussed in Section 3.7.6. The variables needed from program CONVERT are NCOMBS, OVALAP, PADS, W1, W2, W3, DUMMY, and SINGLE. Their part in the fabrication of the glass master will be discussed in the following sections.

5.2.4.3 TAPE29

TAPE29 is usually a temporary disk file (created by program CONVERT). As in the case of TAPE28, TAPE29 contains certain information from program CONVERT. TAPE29 contains the title card of each Ho[N], Ho[NTAP], ACT[M], or TCA[NTAP]deck inputed into program CONVERT. TAPE29 also contains the variables EXTRA, DUMMY, SINGLE, FID, and REV. Only the title cards and the variables FID and REV are read by program HUGHES. FID and REV are both arrays, one value of each array is used for every comb in the SAW device.

REV is a logical array. If REV(I) = .False. then the I'th comb is printed on the glass master in the same order as defined by the

```
RECORD NUMBER       4       MASTER NUMBER       1

    'X271762Y464285SX273532SX368674SX370444SX441580Y156885SX443603SX445625SX447648SY153093SY149300SY1455
    08SY141715SY137923SY134130SY130337SY126545SX449671Y156885SY153093SY149300SY145508SY141715SY137923SY1
    34130SY130337SY126545SX451693Y156885SX453716SX455739SX471920SY153093SY149300SY145508SY141715SY137923
    SY134130SY130337SY126545SX473943Y156885SY153093SY149300SY145508SY141715SY137923SY134130SY130337SY126
    545SX475966Y156885SY145508SY126545SX477988Y156885SY145508SY126545SX480011Y156885SY145508SY126545SX48
    2034Y156885SY126545SX484056Y156885SY126545SX486079Y156885SY126545SX502261Y153093SY149300SY130337SX50
    4283Y156885SY153093SY149300SY145508SY130337SY126545SX506306Y156885SY145508SY126545SX508329Y156885SY1
    45508SY126545SX510352Y156885SY145508SY126545SX512374Y156885SY145508SY126545SX514397Y156885.

RECORD NUMBER       5       MASTER NUMBER       1

    'SY153093SY145508SY141715SY137923SY134130SY130337SY126545SX516420Y153093SY141715SY137923SY134130SY13
    0337SX532601Y156885SX534624SX536647SX538669SY153093SY149300SY145508SY141715SY137923SY134130SY130337S
    Y126545SX540692Y156885SY153093SY149300SY145508SY141715SY137923SY134130SY130337SY126545SX542715Y15688
    5SX544737SX546760SX562942Y153093SY149300SY145508SY141715SY137923SY134130SY130337SX564964Y156885SY153
    093SY149300SY145508SY141715SY137923SY134130SY130337SY126545SX566987Y156885SY126545SX569010Y156885SY1
    26545SX571032Y156885SY126545SX573055Y156885SY126545SX575078Y156885SY153093SY149300SY145508SY141715SY
    137923SY134130SY130337SY126545SX577100Y153093SY149300SY145508SY141715SY137923SY134130SY130337SX59328
    2Y153093SY134130SY130337SY126545SX595305Y156885SY153093SY137923SY134130SY130337SY126545SX597327.

RECORD NUMBER       6       MASTER NUMBER       1

    'Y156885SY141715SY137923SY126545SX599350Y156885SY145508SY141715SY126545SX601373Y156885SY145508SY1265
    45SX603396Y156885SY145508SY126545SX605418Y156885SY153093SY149300SY145508SY126545SX607441Y153093SY149
    300SY125545SZ.
```

Figure 5.1 Tape Records Created by Program HUGHES (After Processing on IBM Equipment)

Each full row is 100 characters long.

AD-A079 623 ANALYSIS AND COMPUTER SYSTEMS INC BURLINGTON MA F/G 17/1
DEVELOPMENT OF NUMERICAL TECHNIQUES AND COMPUTER SYSTEMS FOR DE--ETC(U)
AUG 79 L J ELTERMAN F19628-76-C-0257

UNCLASSIFIED RADC-TR-79-181 NL

2 OF 4
ADA 079623

original deck setup of program CONVERT (i.e. the Ho[N], ACT[M], Ho[NTAP] or TCA[NTAP]decks). If REV(I) = .True. then the I'th comb is reversed on the glass master (see Section 3.4.5.6).

FID(I) is the distance from the left edge of the left finger to the center of the zero'th gap of the I'th comb. NOTE: In an asymmetrical comb FID(I) will vary depending on whether REV(I) is true or false.

5.2.4.4 Namelist $HUGHES

Namelist $HUGHES contains variables that define additional specifications of the glass master. These variables will be discussed in the following sections.

5.2.4.5 Hughes Title Card

The Hughes Title Card goes right before the $HUGHES namelist in the input deck setup. The first 4 characters of this card may be any alphanumeric characters. The 5'th and 6'th characters must be digits. All other characters may be any valid character. The first six characters (actually flashed onto the glass meter) are used as the label name on the glass master. Remaining characters are for user information on the printed output. However, in a special application, program LETTER extracts pertinent characters from the title card for use as described in Section 11.3

5.2.5 Output of Program HUGHES

The output of program HUGHES consists of TAPE49, TAPE50, and printed output.

5.2.5.1 TAPE49

TAPE49 contains the instructions to the Hughes Step and Repeat Machine. TAPE49 is a physical tape as described in Section 5.2.3.

5.2.5.2 TAPE50

TAPE50 is usually a temporary disk file. TAPE50 contains the reticle sizes to be used by the Hughes Step and Repeat Machine in the fabrication of the SAW device. TAPE50 is used by the HUESCHK (Step and Repeat Machine simulation) program. (The reticle sizes are sent to the Hughes Step and Repeat Machine operators via printed output.)

5.2.5.3 Printed Output

The printed output from program Hughes contains:

A) A printout of the records on TAPE49 similar to Figure 5.1.

B) A page containing the reticle sizes to be used in the fabrication of the SAW device. The dimensions specified for these reticles are at the deliverable master size; the mask maker must adjust these upward depending on the reduction factor used in the step and repeat process. TAPE49 and this page are all the Hughes Company needs to fabricate the SAW device.

C) A page containing the final values of important variables in program HUGHES. This page is used by the programmer to help verify the software part of the fabrication process.

D) A page containing a physical description of what the glass master should look like, useful to the people at Hughes and the engineers at RADC, to help verify the final product.

5.2.6 Fabrication Specifications of the Glass Master

5.2.6.1 Introduction

Many of the specifications for fabricating the glass master (usually two inches by two inches), such as the number of fingers per comb, relative OVALAP of fingers, relative distance of fingers from each other, finger widths, etc., come from the Raytheon Format on TAPE9.

Other specifications, such as the actual maximum overlaps of the combs, the number of combs, etc., come from TAPE28, containing the $CONVERT namelist.

The master name label comes from the Hughes Title Card as described in Section 5.2.4.5.

Still other master specifications, such as the placement of the combs on the master, size of the pads, magnification factor, etc., have not yet been discussed.

Some of the remaining specifications are defined according to fixed formulas, others are defined by the $HUGHES namelist.

5.2.6.2 The $HUGHES Namelist Variables and Their Type

PADWID	real
BULGE	real
CONOUT	logical
RET	real
PATHCL	real
MASKCL	real

MASKCL	real
PAD2PAD	real
DIS	real array of length four
FIDWID	real
MASTER	real
KEARNS	logical
SIDEBY	logical
PADONLY	logical
SHORT	logical
METRIC	logical

5.2.6.3 <u>The $HUGHES Namelist Variables and Their Defaults</u>

```
PADWID  =  2.0E-3
BULGE   =  0.
CONOUT  =  .TRUE.
RET     =  3.0E-4
PATHCL  =  2.54E-2
MASKCL  =  2.54E-2
PAD2PAD =  1.27E-3
DIS     =  0.,0.,0.,0.
FIDWID  =  1.27E-3
MASTER  =  10.0
KEARNS  =  .FALSE.
SIDEBY  =  .FALSE.
PADONLY =  .FALSE.
SHORT   =  .FALSE.
METRIC  =  .FALSE.
```

All units are in meters

5.2.6.4 The $HUGHES Namelist Variables and Their Functions

MASTER: The size of the master image and the size of the final physical SAW device are not usually the same and, in fact, differ by a factor of "MASTER". MASTER is usually set to ten and is used to adjust the incoming parameters from CONVERT (always specified at the final device size) to the correct dimensions for the master.

PADWID: There are two pads for every Comb. There are as many as five Combs per SAW device made from program HUGHES. Every pad is the same width. This width is "PADWID". PADWID values are absolute and not multiplied by the variable master, i.e. the width of the pads on the glass master is simply "PADWID".

BULGE: When the length of a finger on a transducer is longer than the value RET (see below) then the finger must be made with more than one flash. "BULGE" is the amount the finger flashes overlap. A BULGE value of 0.0 (no overlap) has been found to work fine.

CONOUT: This is an obsolete variable. The user should always let CONOUT = .TRUE. (default value). If CONOUT = .FALSE. then both fingers in a double electrode pair are connected together by a thin metal strip (see Figure 5.2).

RET: This is the length the user chooses to make the reticle that flashes the fingers. The RET value almost always must be input by the user, the default value is rarely valid. If the user chooses too small a value of RET then the fingers will have to be made with more than one flash. Choosing a small RET value does not necessarily allow complete flexibility in producing a finger. Finger lengths will always be of the form

(NFLASHES*RET) - ((NFLASHES-1) * BULGE)

where NFLASHES is the number of flashes used to make the finger. Extra unnecessary flashing may result in added expense and loss of accuracy.

If the user chooses too large a value of RET, the inside edges of the fingers are lined up according to specifications from TAPE9 while the outside edge falls

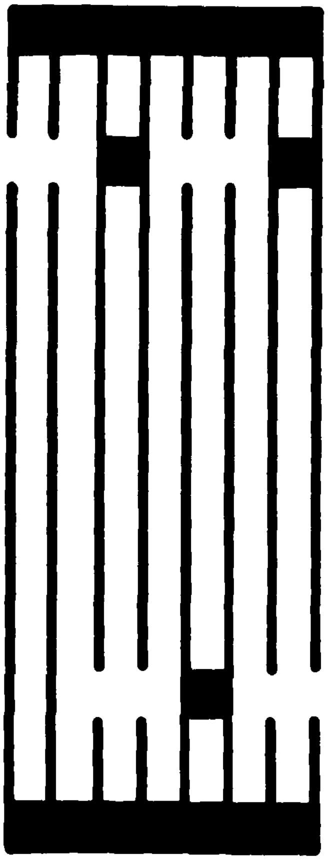


Figure 5.2 Double Electrode Comb with CONOUT = .False.

wherever it may fall, often beyond the PAD. In the case of an apodized transducer (a transducer with varying finger lengths), the user may choose to use the longest finger length as his value of RET even though this may cause the shorter fingers' outside edges to fall beyond the PAD (see Figure 5.3). Most of the time this does no harm. In the case of a non-apodized transducer, or when the engineer does not care if the fingers extend beyond the PAD, then the following value of RET is usually used: (OVALAP and PADS from the $CONVERT namelist).

RET = MASTER*(OVALAP + 2*PADS)

PADS + OVALAP equals the length of the longest finger on the finished physical transducer. The extra length "PADS"makes sure the longest finger is firmly implanted in the PAD where roundoff error will not cause an extra flash. Multiplying it by MASTER equates the proper reticle size to the glass master.

PATHCL: PATHCL specifies the vertical distance (along the Y-Axis) from the origin to the center of the glass master. Thus, the vertical length of the glass master is 2*PATHCL. Since RADC engineers almost always use two inch square masters the usual value for PATHCL is 2.54E-2 (meters) (see Figure 5.5).

MASKCL: MASKCL specifies the horizontal distance (along the X-Axis) from the origin to the center of the glass master. Thus, the horizontal length of the glass master is 2*MASKCL. Since RADC engineers almost always use two inch square masters the usual value for MASKCL is 2.54E-2 (meters) (see Figure 5.5).

PAD2PAD: Assuming DIS(1) = DIS(2) = DIS(3) = DIS(4) = 0 (default, see following for definition of DIS), then the horizontal distance between any two consecutive combs on a master (from PAD to PAD) is "PAD2PAD". In other words, all combs on the glass master are evenly separated by distance PAD2PAD (see Figure 5.5). If NCOMBS = 1, PAD to PAD is ignored.

DIS(I): The DIS array is capable of overriding the variable PAD2PAD. If DIS(I) ≠ 0, then the right edge of the rightmost finger of the I'th comb is separated from the left edge of the leftmost finger of the (I+1)'th comb by the distance DIS(I). This allows varying distances between the combs (see Figure 5.5).

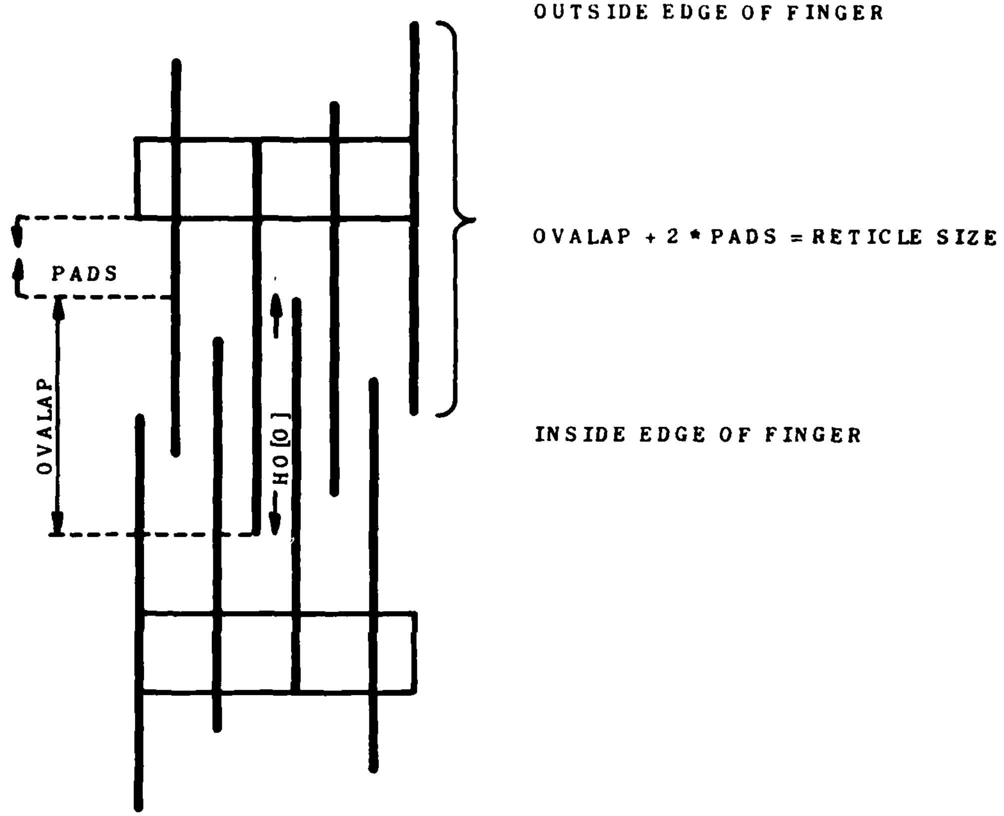


Figure 5.3 Fabrication of Apod:zed Transducer with Large Reticle

Assume MASTER = 1 (i.e., the transducer on the glass master is the same size as the final physical transducer). The figure above represents an apodized transducer (varying finger ovalaps). A large value of RET was chosen causing most of the fingers to be too long. The inside edges of the fingers are line up accurately. The outside edges are forced to extend beyond the Pads. Note that in actuality the whole Pad is uniform clear (in the case of the negative glass master) or uniform metal (in the case of the final product). The contrast where the fingers extend into the Pad in the above figure is for illustrative purposes only.

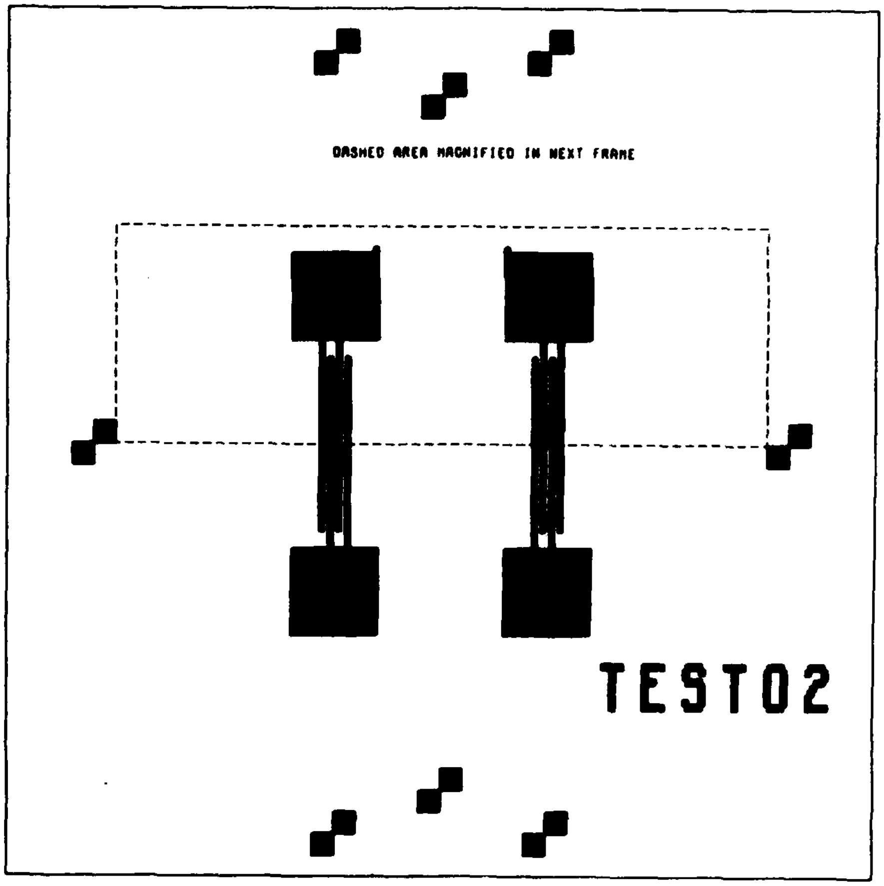


Figure 5.4 A Plot of a Glass Master Produced By the Simulation Program HUESCHK

The actual master is two inches by two inches. Everything shown in this plot except the dotted lines and the words "dashed area magnified in next frame" is actually flashed onto the glass master. The small tic mark on the Pads identifies the NMIN'TH side of the transducer. This is necessary for slightly non-symmetric transducers when only one is on a master.

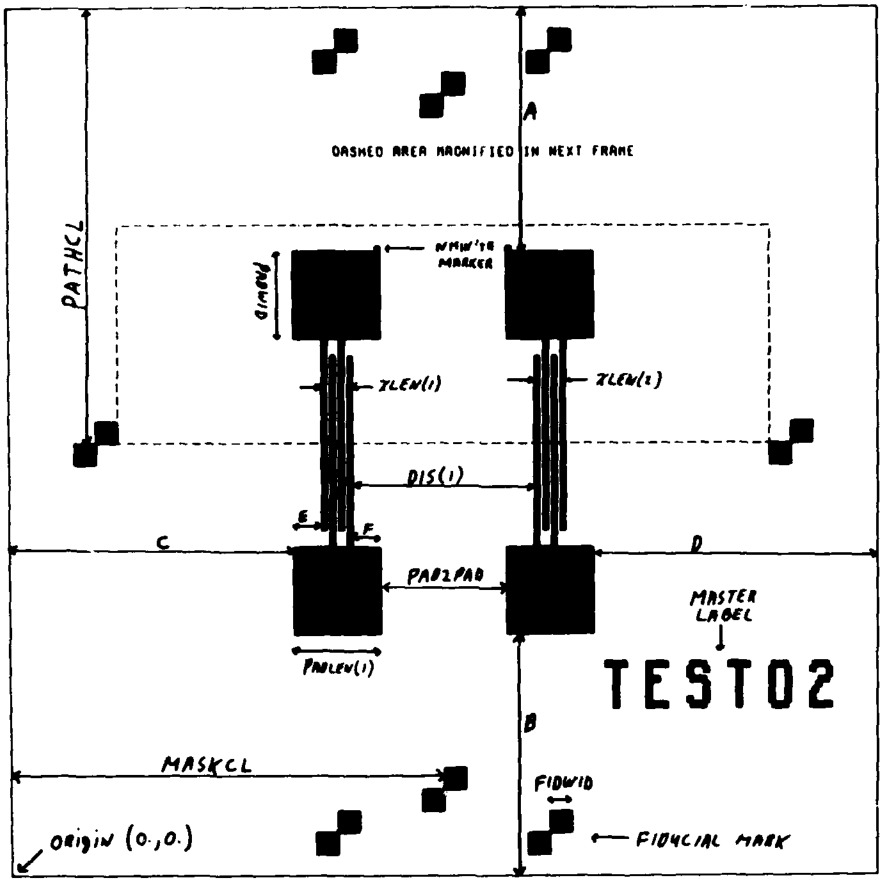


Figure 5.5 A Plot of a Glass Master Produced by the HUESCHK Simulation Program

The actual Glass Master is two inches by two inches. Definitions are shown as defined in Section 5.2.6. Also,

Distance "A" = Distance "B" (See Section 5.2.6.3.2)
Distance "C" = Distance "D" (See Section 5.2.6.3.1)
Distance "E" = Distance "F" (See Section 5.2.6.3.4)

FIDWID: FIDWID is the width and length of the square fiducial marks. Fiducial marks will be explained in following text (see Figure 5.5).

KEARNS: If KEARNS = .TRUE., then each comb is centered on a separate master. Example: If NCOMBS = 3 and KEARNS = .TRUE. then three different masters will be created, each with one comb. The magnetic tape processed at MITRE would then generate three individual tapes for shipment to HUGHES. The above example is equivalent to the user running Hughes three separate times, each run containing the specifications for one of the combs, and taking three separate tapes to MITRE. When KEARNS = .TRUE. each master is fabricated complete with fiducial and NMIN'th markers (see following text).

SIDEBY: If SIDEBY = .TRUE., then the combs are stacked side by side on the glass master with the 0'th gap of the stacked combs lining up vertically. The distance between the pads of the stacked combs separated by distance PAD2PAD. The set of combs will be centered both horizontally and vertically on the glass master. That is the horizontal distance from the left edge of the glass master to the center of the 0'th gap will be equal to the horizontal distance from the center of the zero'th gap to the right edge of the glass master, and the vertical distance from the bottom of the bottommost pad to the bottom edge of the glass master will be equal to the vertical distance from the top of the topmost pad to the top of the glass master (see Figure 5.6).

PADONLY: If PADONLY = .TRUE. then the flashing of the fingers is supressed on the glass master, all other specifications remain the same (see Figure 5.7).

SHORT: If SHORT = .TRUE. then all the area between the pads (where the fingers usually go) become shorted out (with clear on the negative master, or metal on the final product (see Figure 5.8).

METRIC If METRIC = .FALSE., then the units given on the magnetic tape sent to Hughes are in laser counts (321103 per inch). If METRIC = .TRUE., then the units are in measures of 10^{-7} meters. That is, 1 unit = 10^{-7} meters.

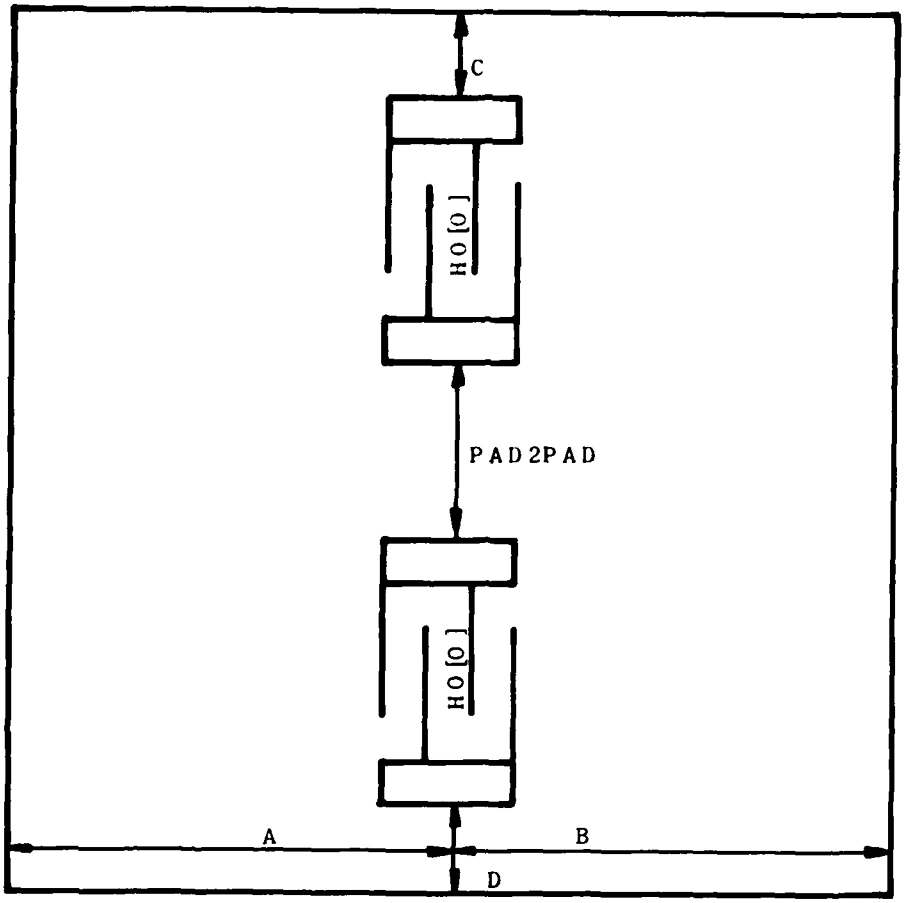


Figure 5.6 Simplified Diagram of a Two Comb Master where SIDEBY = .True.

Distance A = Distance B and
Distance C = Distance D as described
in Section 5.2.6.4 under "SIDEBY"

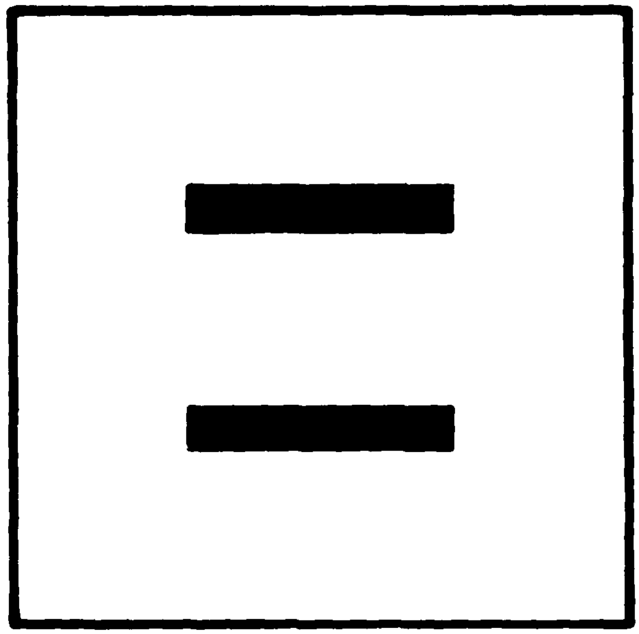


Figure 5.7 Simplified Diagram of One Comb
Master where PADONLY = .True.

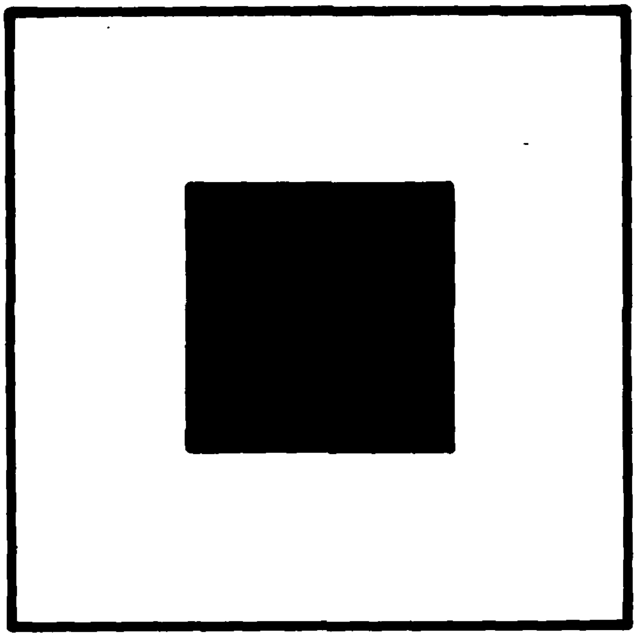


Figure 5.8 Master With SHORT = .True.

Simplified Diagram of the same one comb master shown in Figure 5.7, only

PADONLY = .False. and
SHORT = .True.

5.2.6.5 Other Specifications Not Directly Defined by the $HUGHES Namelist or TAPE9

5.2.6.5.1 Horizontal Centering of Combs on the Glass Master

The distance from the left edges of the pads of the leftmost comb to the left edge of the glass master will be equal to the distance from the right edges of the pads of the rightmost comb to the right edge of the glass master (see Figure 5.5).

5.2.6.5.2 Vertical Centering of Combs on the Glass Master

Since the values of OVALAP, PADS, and PADWID are the same for all combs on a master, it follows that the total widths of all combs on a master are equal. All combs on a master are lined up vertically, that is, the distance from the top of the top pad to the top of the master and the distance from the bottom of the bottom pad to the bottom of the glass master is the same for all combs on the master (see Figure 5.5).

5.2.6.5.3 Pad Length of the I'th Comb

PADLEN(I), the length of the pads on the I'th comb, are calculated in the following way:

Let XLEN(I) = the distance from the left edge of the leftmost finger of the I'th comb to the right edge of the rightmost finger of the I'th comb.

PADLEN(I) = the maximum of XLEN(I) and PADWID (see Figure 5.5).

5.2.6.5.4 Centering of Fingers on Pads

For all combs on a master the horizontal distance from the left edge of the pad to the left edge of the leftmost finger will be equal to the horizontal distance from the right edge of the rightmost finger to the right edge of the pad. This distance will be zero if XLEN(I) $\geq$ PADWID (see Section 5.2.6.3.3 and Figure 5.5).

5.2.6.5.5 Fiducial Marks

A fiducial mark is made up of two squares, where the squares touch represents a point. Fiducial marks come in pairs. The purpose of the fiducial marks is to allow the device engineer (at RADC) to line up the glass master on a machine prior to the projection process of producing a real transducer from a glass master. Lining up the points represented by one set of fiducial marks, separates the glass master into a top half and a bottom half. Lining up the points, represented by another set of fiducial marks, separates the master into a left and a right half. There are also a set of

fiducial marks for each comb on the master. These are called the O'th gap fiducials. If one were to draw a line through the points a particular set represents, the line would go through the middle of a zero'th gap (see Figure 5.5).

5.2.6.5.6 NMINTH Marks

The tiny squares on the top pads of the combs show whether or not the particular comb has been reversed from the original data (see Sections 5.2.4.3 and 3.4.6.3). If the NMINTH marker is on the left side of the pad then that comb has not been reversed. (REV(I) = .FALSE.). If the NMINTH marker is on the right side of the pad, then the comb has been reversed (REV(I) = .TRUE.) (see Figure 5.5).

5.2.6.5.7 Master Label

A Master Label name is flashed onto the lower right corner of all masters. This name is obtained from the Hughes Title Card (see Section 5.2.4.5 and Figure 5.5).

5.2.6.5.8 Finger Widths

Every finger on a glass master fabricated from program HUGHES has the same width. Although the option to have varying finger widths was built into program CONVERT, it is not implemented into program HUGHES. This design decision was made for two reasons.

A) Fabricating a finger from more than one flash may result in loss of precision.

B) Changing reticles is costly (each reticle has to be precision cut) and changing reticles may result in loss of precision.

It is impossible to have varying finger widths without using either procedure A or procedure B.

The exact position of a finger on a master is calculated as follows:

A) The X-coordinate of the left edge of a finger is obtained from the Raytheon Format on TAPE9 (see Section 3.7).

B) The X-coordinate of the right edge of a finger is obtained by taking the X-Left-Coordinate and adding the Value W1(1). W1 is from the $CONVERT namelist and is passed to Hughes via TAPE28 (see Section 5.2.4.2). These relative coordinates are then modified in accordance with the magnification factor "MASTER", to produce the glass master. All the fingers of the final device, of course, have width W1(1).

5.2.7 Reticle Changes

During the fabrication procedure of a glass master from a magnetic tape produced by program HUGHES, the reticle is normally changed three times. The reticle changes are as follows:

Finger Producing Reticle	Original Reticle
PAD Producing Reticle	First Change
Fiducial Producing Reticle	Second Change
Label Producing Reticle	Third Change

The HUGHES program automatically puts the Z command at the right places on the magnetic tape to allow the operator to change reticles (see Section 5.2.3).

The HUGHES program also calculates the most efficient reticle sizes for the four above tasks (except for the finger reticle whose length is input by the user), and prints out an information sheet telling the operators what size reticles to "cut out" and after which pauses to use them.

Program HUGHES does not allow the user to specify dimensions that will result in the use of a reticle wider than 2.5×10^{-3} meters or longer than 5.08×10^{-2} meters.

5.3 Program HUGHESXY

5.3.1 Introduction

Program HUGHESXY is basically a more versatile version of program HUGHES. The added features of program HUGHESXY are as follows:

A) Program HUGHESXY allows for up to sixteen combs on a master. Program HUGHES allows only a maximum of five combs on a master.

B) Program HUGHES allows the user only limited flexibility in the placement of the combs on the master. Program HUGHESXY allows the user complete flexibility in the placement of the combs on the master.

C) Program HUGHES does not allow the user to vary the widths of fingers on a master. All fingers on a master are the same width. Program HUGHESXY allows the user to vary the finger widths from comb to comb on a master (all fingers within a comb, however, must have the same width).

D) All combs on a master produced by program HUGHES have the same width pads as each other, this width being determined by the variable PADWID. Program HUGHESXY allows more flexibility in determining pad widths. Both pads on a single comb must still be the same width; however, the pad widths may vary from comb to comb. The pads of a particular comb can even be suppressed entirely if the user so desires.

E) The pad lengths in program HUGHES are determined by a set formula (see Section 5.2.6.3.3). Program HUGHESXY allows the user almost total flexibility in determining the pad length.

F) All combs on a master produced by program HUGHES have the same maximum overlap. Program HUGHESXY allows varying maximum overlaps from comb to comb.

G) The distance PADS (see Section 3.6.4) is the same for all combs on a master produced by program HUGHES. Program HUGHESXY allows the variable PADS to vary from comb to comb.

The input to program HUGHESXY consists of TAPE9, TAPE28, TAPE29, the $HUGHES namelist and the HUGHESXY Title Card. This is the same as for program HUGHES, only with program HUGHESXY TAPE9, TAPE28, and TAPE29 must come from program CONVXY, and not Program

CONVERT, since the formats of these tapes are not exactly the same. (For example TAPE28 contains the $CONVERT namelist which is not the same for both CONVXY and CONVERT.) Also note that the $HUGHES namelist is not the same for program HUGHES as it is for HUGHESXY.

5.3.2 Added or Modified $HUGHES Namelist Variables Supplying Additional Versatility

NOPAD: NOPAD is an added $HUGHES namelist variable. NOPAD is a logical array. If NOPAD(I) = .TRUE., then the I'th comb on a master will be fabricated with no pads. That is, only the fingers will be fabricated. This is true no matter what value PADWID(I) has (see next definition). Default Values: NOPAD(I) = .FALSE. for all combs.

PADWID: In program HUGHESXY PADWID is a real array. If, for the I'th comb

PADWID(I) > 0 and NOPAD(I) = .FALSE.

then the I'th comb will have two pads, each having width PADWID(I).

If PADWID(I) ≤ 0 then NOPAD(I) will be set to TRUE, and the I'th comb will have no pads.

Default Value: PADWID(I) = 2.0E-3 for all combs.

RET: In program HUGHES RET is a real variable. In program HUGHESXY RET is a real array. In program HUGHES the user could only choose one reticle length to produce all the fingers on a master. Program HUGHESXY allows the user to choose a different reticle length (for producing the fingers) for each comb on a master. Within a comb the reticle length must be constant, this length being RET(I). Within a comb the same rules apply to RET as those discussed in Section 5.2.6.4.

Default Value for RET(I) = MASTER * (2*PADS(I) + OVALAP(I))

(PADS(I) and OVALAP(I) come from CONVXYS' $CONVERT namelist. Note that unlike program HUGHES the default values for RET are meaningful.)

PADLEN: PADLEN is an added $HUGHES namelist variable. PADLEN is a real array which allows the user to choose the pad lengths for any comb on the glass master. Both pads on a particular comb, however, must be the same length. The actual length of the pads on the I'th comb of a master is

PADLEN(I) * MASTER. Note that this is the only exception to the rule of entering $CONVERT data at device size and $HUGHES data at master size. (For example, the PADWID values are absolute and not multiplied by the variable master, i.e. the width of the pads of the I'th comb on the glass master is simply PADWID(I).)

If no value for PADLEN(I) is specifically inputed by the user (via namelist $HUGHES) then a temporary default value of PADLEN(I) = .9E+11 is assigned. This enormous value for PADLEN(I) indicates, that although the pad length is not available at the time the $HUGHES namelist variables are printed out, eventually the pad length will be calculated in the standard way (discussed in Section 5.2.6.3.3).

If for a particular comb, the user defined value of PADLEN(I) is less than XLEN(I) (see Section 5.2.6.3.3), then PADLEN(I) is set to XLEN(I). That is, PADLEN(I) cannot be less than XLEN(I).

XP:
YP: XP and YP are both added $HUGHES namelist variables. Both XP and YP are real arrays. The value of XP(I) and YP(I) determine the position of the I'th comb. The I'th comb is positioned according to the center of its zero'th gap (by the "center" of the zero'th gap we mean the exact center both horizontally and vertically).

Example: XP(3) = 2.54E-2, YP(3) = 2.54E-2. This means the center of the zero'th gap of the third comb will be placed one inch above, and one inch to the right of the origin on the glass master. Default values are seldom used, the user must almost always input these arrays.

Defaults: XP(I) = .002988I
YP(I) = .0254

5.3.3 Other $HUGHES Namelist Variables Used in HUGHESXY

BULGE:	As in Program HUGHES
CONOUT:	As in Program HUGHES
PATHCL:	No longer used
MASKCL:	No longer used
PAD2PAD:	No longer used
DIS:	No longer used
FIDWID	As in program HUGHES
MASTER:	As in program HUGHES

KEARNS:	KEARNS must equal FALSE (default)
SIDEBY:	SIDEBY must equal FALSE (default)
PADONLY:	As in progrm HUGHES
SHORT:	As in program HUGHES
METRIC:	As in program HUGHES

5.3.4 Variables From The $CONVERT Namelist Supplying Additional Versatility

OVALAP: The OVALAP values are supplied to program HUGHESXY via TAPE28. If TAPE28 has been created by CONVXY (as it must be to run HUGHESXY) then OVALAP will be a real array. The maximum overlap of the fingers of the I'th comb will be MASTER*OVALAP(I) (on the glass master).

PADS: The PADS values are supplied to program HUGHESXY via TAPE28. If TAPE28 has been created by CONVXY (as it must be to run HUGHESXY) then PADS will be a real array (see Section 3.6.4 for a definition of PADS). The values of PADS will be multiplied by MASTER to create the glass master.

NCOMBS: In program HUGHES the maximum value for NCOMBS is five. In program HUGHESXY the maximum value is sixteen.

W1: W1 is a real array. In program HUGHES all the combs of a glass master have the same width, namely W1(1)*MASTER. In program HUGHESXY the widths of the fingers in the I'th comb is W1(I)*MASTER.

5.3.5 Reticle Changes

Program HUGHES uses only one reticle size to fabricate all the fingers on a given master. Program HUGHESXY is capable of producing different combs with different finger widths and different reticle lengths. Thus, it is possible that a different reticle may have to be used for each comb on the master.

The I'th comb on glass master will be created with a new reticle if:

$$W1(I) \neq W1(I-1)$$

or

$$RET(I) \neq RET(I-1) \quad \text{For all I such that } 2 \leq I \leq 16$$

Since the user has complete control over the positioning of any comb the deck should be set up so that all combs using the same finger reticles are processed successively. This eliminates unnecessary reticle changes.

5.3.6 TAPE48

TAPE48 is created by program HUGHESXY. TAPE48 contains the same information as TAPE49. The only difference is that TAPE49 is usually a physical tape and TAPE48 is always a disk file. TAPE48 is used as input to program HUESCHKTEK (see Section 5.5).

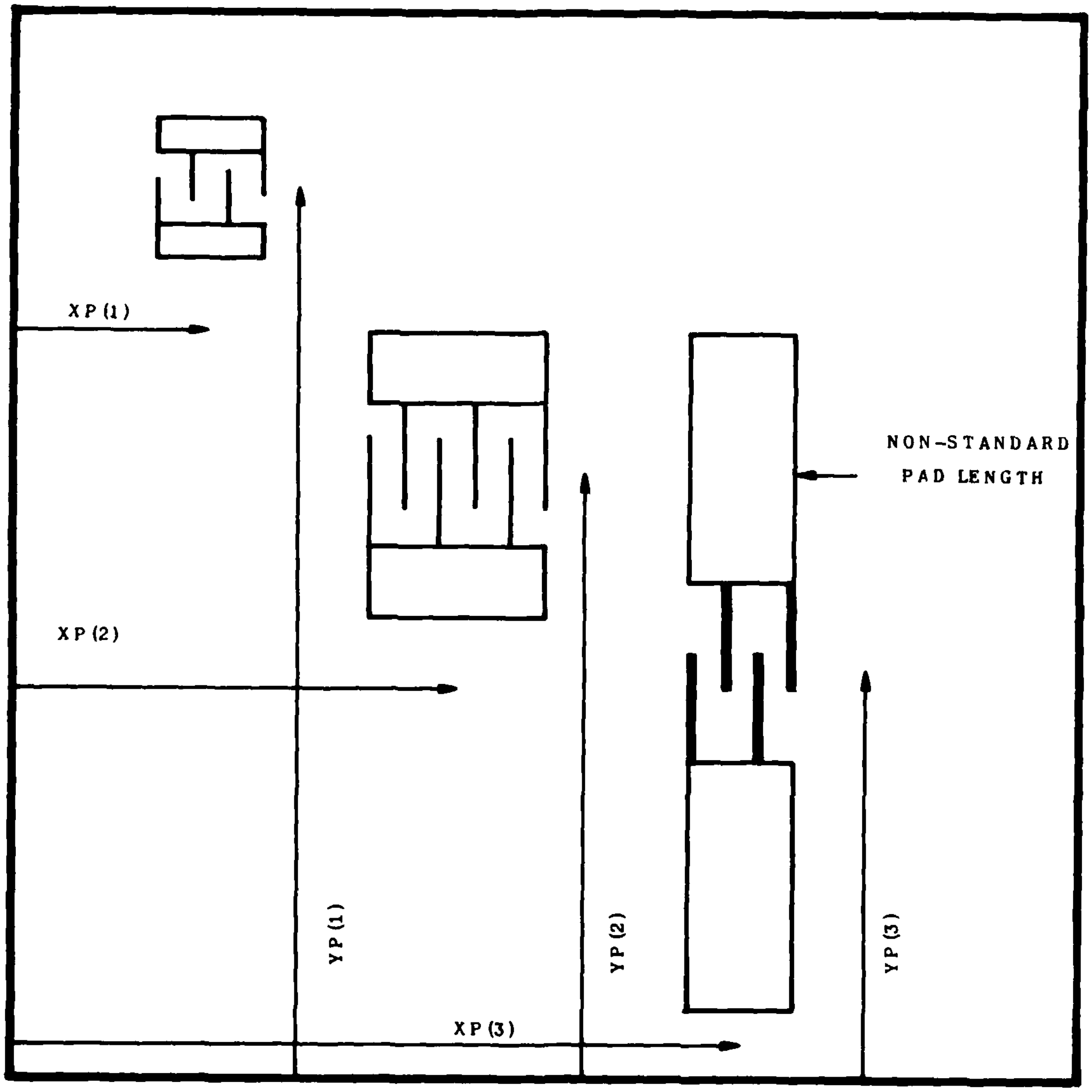


Figure 5.9 Simplified Diagram of a Master
Produced by Program HUGHESXY

Note varying values for OVALAP, PADS and PADWID.
Note also that Comb 3 has wider fingers than the others.

5.4 Program HUESCHK

5.4.1 Introduction

Program HUESCHK is a simulation program. HUESCHK simulates the Hughes Step and Repeat Machine. Input to the program is TAPE49 from either program HUGHES or HUGHESXY. Output of the program is a plotted picture of the glass master the Hughes Step and Repeat Machine is expected to fabricate.

5.4.2 Input to Program HUESCHK

5.4.2.1 TAPE49

TAPE49 is a physical tape which is created by either program HUGHES or program HUGHESXY.

5.4.2.2 TAPE50

TAPE50 is a temporary disk file created by program HUGHES or HUGHESXY. TAPE50 contains the reticle sizes to be used by the Hughes Step and Repeat Machine in the fabrication of a SAW device. TAPE50 is used only by programs HUESCHK and HUESCHKTEK. In the actual fabrication of the glass master the reticles are changed by the operator who gets the information (what size reticle to use, when to change them) from printed output created by program HUGHES or HUGHESXY.

5.4.2.3 HUESCHK Title Card

The card input to program HUESCHK consists of an optional HUESCHK title card followed by the optional $HUESCHK namelist. The first 30 characters of the HUESCHK title card are printed on the outputted plots and should contain the name and telephone number of the user.

5.4.2.4 The $HUESCHK Namelist

5.4.2.4.1 The $HUESCHK Namelist Variables and Their Type

PLTSIZE:	real
MSTSIZE:	real
MAG	real
METRIC	logical
PDEN	real
FDEN	real

HALF	logical
KPIK	integer

5.4.2.4.2 <u>The $HUESCHK Namelist Variables and Their Defaults</u>

```
PLTSIZE  =  10.
MSTSIZE  =   2.
MAG      =   8.
METRIC   =  .False.
PDEN     =   8.
FDEN     =  75.
HALF     =  .True.
KPIK     =   0
```

5.4.2.4.3 <u>The $HUESCHK Variables and Their Defaults</u>

PLTSIZE: The size of the plotted simulation of the glass master is PLTSIZE inches long by PLTSIZE inches wide. (It is assumed that the glass master is square.)

MSTSIZE: MSTSIZE is the length and width (in inches) of the actual master to be fabricated. (It is assumed that the glass master is square.)

MAG: Figure 5.4 is an example of a plotted picture produced by program HUESCHK. Note the rectangle outlined by the dotted line. This portion of the picture will be magnified in the next plotted frame. The magnification will be such that the width of the rectangle will take up the whole width of the paper. (That is, the width will be "PLTSIZE", which is usually ten inches.) The bottom edge of the rectangle always goes from fiducial mark to fiducial mark. The height of the rectangle will be:

PLTSIZE/(2*MAG) inches.

In the example shown in Figure 5.4, MAG = 2. Thus, the height of the rectangle is PLTSIZE/4, and the rectangle will be magnified four times in the next frame. In general the magnification factor will be (2*MAG). If MAG = 0.0, then the magnification frame is suppressed.

METRIC: The value of METRIC in the $HUESCHK namelist should always be the same value as METRIC in the $HUGHES namelist. (See Section 5.2.6.4.)

PDEN: In Figure 5.4 one can see that the pads are colored in (represents uniform clear on the glass master). This coloring in process is accomplished by the plotter drawing many fine lines (in both the vertical and horizontal direction). The number of lines drawn per inch is PDEN. Note that since program HUESCHK is a simulation of the Hughes Step and Repeat Machine, coloring is done flash by flash. Thus, if a pad has been created with overlapping reticle flashes it will appear that some areas of the pads on the plotted picture are denser than other areas (because they have been colored in twice). On the actual master, however, the pad will be uniform clear. This is because an area flashed twice is no more clear than an area flashed only once.

Example: A square pad 1/4 inch by 1/4 inch has been created with a single flash. If the plotted simulation picture has been created with PDEN = 40, then the plotter will create this pad by first drawing the outline and then filling it in with ten lines in the horizontal direction and ten lines in the vertical direction.

Some HUESCHK plots take a very long time to plot. The purpose of PDEN is to allow the user to choose the best trade off between plot detail and the time it takes to produce the plot.

FDEN: Similar to PDEN, only used in the plotting of the fingers. Fingers are plotted with only vertical lines. FDEN is the number of vertical lines used per inch to plot a finger.

HALF: In Figure 5.4, HALF = .False.. If HALF = .True. then the right half of the rectangle will not be magnified. (The actual drawing of the dotted rectangle will be half of that shown in Figure 5.4.) In the case of a symmetrical master the right half is the same as the left half and does not need to be magnified. HALF = .True. saves plotting time.

KPIK: In creating a plot the CDC software creates a set of machine plotting instructions which go on magnetic tape. This magnetic tape is then fed to the CAL-COMP plotter which draws the plot. If the plot is very long and complex (as some HUESCHK plots are) it may take more than

one magnetic tape to hold all the plotting instructions. It has been found that when this is the case, the resulting plot is much more likely to be faulty than when all the plotting instructions go on a single tape. For this reason the capability to have program HUESCHK plot only one of the combs on a master has been designed into the program logic.

If KPIK = 0, all combs on the master will be plotted

If KPIK = 1, only the first comb on the master will be plotted.

If KPIK = N, only the N'th comb on the master will be plotted.

Example of Use: You are fabricating a three comb master. Each comb on the master is very complex and the plotting instructions to plot any of the three combs take up almost a full reel of magnetic tape. To plot the whole master all at once requires three magnetic tapes, thereby greatly increasing the odds of a hardware plotting failure. Not only that, but because it takes so long to plot the three comb master all at once, the time required to receive the plot is increased significantly.

Solution: Save TAPE49 (TAPE50 is a physical tape) on permanent disk file. The first day run HUESCHK with KPIK = 1. After receiving the plot, run HUESCHK with KPIK = 2. The third run let KPIK = 3. In this way all plots will be done on their own tape thereby reducing the odds of hardware plotting failure. The odds of success are increased and even if there is a plotting failure on one of the runs, only that run will have to be resubmitted. If you choose KPIK = 0, and a plotting error occurs the whole run must be resubmitted. (Three combs to be replotted instead of one.) To get an idea of how long some of these plots can take to complete, note that some of the complex plots could take up to fourteen hours to complete if the economy measures outlined above were not used.

It is also possible to have all three combs plot on three different tapes (exactly one comb per tape) during a single run, but this increases the chance of operator error as they are not used to changing plot tapes until they are completely full. (See appendix for details of the run.)

5.4.3 Output of Program HUESCHK

The output of program HUESCHK is the plotted picture of the glass master and printed output giving the values of the $HUESCHK variables.

5.5 Program HUESCHKTEK

5.5.1 Introduction

Program HUESCHKTEK is similar to program HUESCHK. The difference is, that with program HUESCHK the picture of the master is plotted on a CAL-COMP plotter. With program HUESCHKTEK the picture of the master is shown on the cathode ray tube of the Tektronix Machine. By pushing a button on the Tektronix Machine the user may obtain a hard copy of the picture shown on the screen. The advantage of using the Tektronix Machine is that the turn-around time is immediate. When using program HUESCHK the user must sometimes wait as long as two or three days to receive his plot. The user may get a picture on the Tektronix Machine by using program HUESCHKTEK immediately after program HUESXY has been run.

The disadvantages of using HUESCHKTEK are:

A) The quality of the picture (size and resolution) is much poorer on a hard copy produced by HUESCHKTEK than on a plot produced by program HUESCHK.

B) The magnified frame often comes out distorted on the Tektronix Machine due to its limited screen size.

It is possible to run program HUGHESXY and HUESCHK in the same run and then go over to the Tektronix Machine and run program HUESCHKTEK. In this way the user can know immediately if the master being simulated looks reasonable. If the Tektronix picture is not what the user expected one can go to the Computer Center and abort one's plot job thereby saving valuable plot time. If the Tektronix picture seems reasonable, one can wait for the CAL-COMP plot for further details. Program HUESCHKTEK is particularly useful in debugging a new version of program HUGHES or HUGHESXY. In order to use the Tektronix Machine it is advised that the user know something about the Hanscom AFB CDC 6600 time-sharing system, otherwise one may run into situations one does not know how to handle. (Time limits, typing mistakes, etc.)

5.5.2 Input to Program HUESCHKTEK

Input to program HUESCHKTEK consists of TAPE49, TAPE50 and input typed directly into the Tektronix Machine.

5.5.2.1 TAPE49

Describing where TAPE49 comes from can be somewhat confusing. First, consider program HUGHESXY. Program HUGHESXY produces both TAPE48 and TAPE49. TAPE49 is a physical tape. A physical tape cannot be read from the Tektronix Machine. For this reason TAPE48 has been created. TAPE48 contains the same information as TAPE49, only it is a disk file instead of a physical tape. Thus, if the user is planning to run program HUESCHKTEK from the Tektronix Machine, TAPE48 must be saved as a permanent file.

Now, consider it from the point of view of program HUESCHKTEK. HUESCHKTEK requires a disk file called TAPE49 as input. Thus, the user must create a TAPE49 from the information saved as TAPE48. The details for doing this are described in Sections 5.5.3 and 5.5.4.

TAPE48 has not yet been implemented into Program HUGHES. However, this is a very easy modification. Simply change the "PACKIT" subroutine in program HUGHES to look like the "PACKIT" subroutine in program HUGHESXY.

5.5.2.2 TAPE50

TAPE50 is created by HUGHESXY (or HUGHES) and contains the reticle sizes needed for the simulation. (See Section 5.2.5.2.)

5.5.2.3 User Input Directly Into the Tektronix Machine

This will be discussed in Section 5.5.4.

5.5.3 Preliminary Steps Needed to Run Program HUESCHKTEK

In the control card setup, before program HUGHESXY (or in the future program HUGHES) is run, request TAPE48 and TAPE50 as permanent files. After program HUGHESXY has been run save TAPE48 and TAPE50 as permanent files. An example of this section of the control cards follow:

```
REQUEST,TAPE48,*PF.

REQUEST,TAPE50,*PF.

HUGHSXY.

CATALOG,TAPE48,SAVETAPE48,ID=ELTERMAN,MR=1

CATALOG,TAPE50,SAVETAPE50,ID=ELTERMAN,MR=1.
```

5.5.4 Running Program HUESCHKTEK on the Tektronix Machine

Assuming the preliminary steps discussed in Section 5.5.3 have been followed, complete the following dialog with the machine. In the following example, underlined text represents text written on the Tektronix Machine by the computer. Non-underlined text represents commands typed into the Tektronix Machine by the user. Text in parenthesis represents the author's comments. ↓ means the user should press the carriage return on the Tektronix Machine.
"Δ" is the notation used to mean the user should press the space bar.

LOGIN,NAME,CODE,861444,SUP ↓
(The user must have one's own LOGIN, Name and Code)

COMMAND SCREEN,80,66.↓

COMMAND ATTACH,HCHECK,HUESCHKTEK,ID=ELTERMAN,MR=1. ↓

(HUESCHKTEK must be cataloged under user's ID as a Binary file)

COMMAND ATTACH,TAPE49,SAVETAPE48,ID=ELTERMAN,MR=1. ↓

COMMAND ATTACH,TAPE50,SAVETAPE50,ID=Elterman. ↓

COMMAND ATTACH,TEK,TEKLIB. ↓

COMMAND LIBRARY,TEK. ↓

(Wait for the computer to printout the following message)

PLEASE INPUT HUESCHK NAMELIST CARD

Δ $HUESCHK$ ↓ (Options may be inputed if the user wishes)

(The computer will now print out some of the namelist variables. After this the computer will stop and wait for the user to press the carriage return.)

(The computer will now write the following message:)

1 GO/ERASE
2 GO/SAVE
3 EXIT
ENTER OPTION

(The user should respond with:)

1↓

(The above message may be repeated several times; each time the user should type:)

1↓

(Until a picture similar to Figure 5.10 comes on the screen. At this point the user can study the picture and get a hard copy if he wishes by pressing the "Copy" button. (Make sure the hard copy machine is turned on and warmed up. After leaving, turn the hard copy machine off.) The user may type:

1 ↓

(Once more to see the (probably distorted) magnified frame. After this type:)

3 ↓

<u>COMMAND</u> LOGOUT

(Push the screen reset button and you are now ready to leave.)

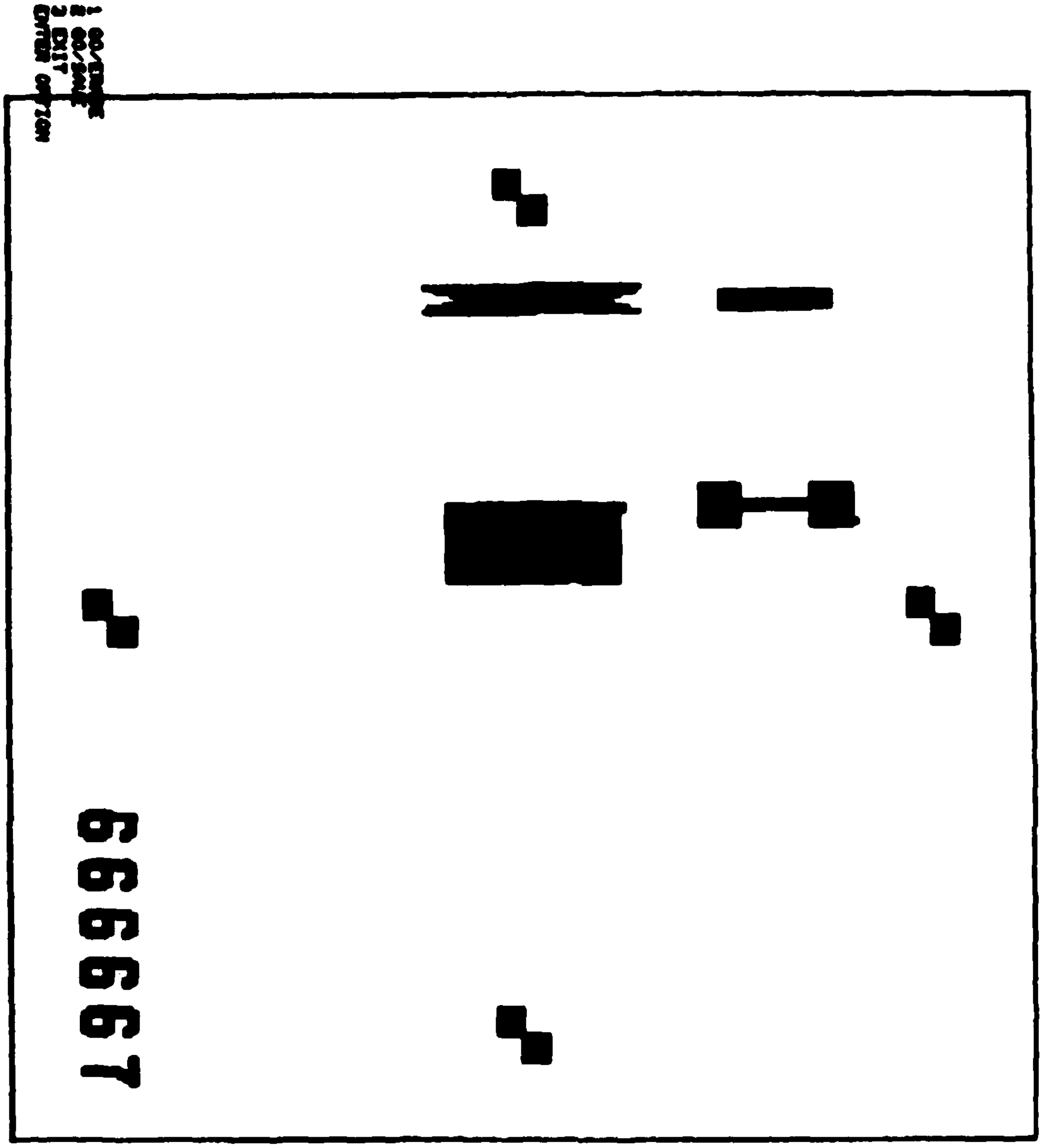


Figure 5.10 HUESCHKTEK Picture

Picture of master created by HUESXY. Picture was created by Program HUESCHKTEK on the TEKTRONIX machine.

6.0 SPECIFYING A SAW DEVICE MASTER (PATTERN GENERATOR)

6.1 Introduction

The programs described in this chapter were originally designed to interface with equipment and procedures at Electromask, Inc. and the Sperry Univac Company. However, these programs should be able to be used, with little or no modification to utilize any Electromask Pattern Generator. The purpose of these programs is to produce printed output, plotted output and magnetic tapes that allow Sperry, for example, to produce a glass master, and to allow both Sperry and the device engineers to verify the correctness of the finished master.

The programs described in Chapter 5 are very similar to the programs discussed in this chapter, thus, it is mainly the differences between the two sets of programs that will be discussed in this chapter. For a thorough understanding of the fabrication procedure Chapter 5 (which will be referenced many times in this chapter) should be read first.

The steps in producing a finished SAW device (Surface Acoustic Wave Device) with, for example, the Sperry Univac Company are as follows:

A) The device engineer describes the device using one of the RADC formats (see Chapter 3, program CONVERT).

B) This format is used as input to program CONVERT which produces as output a description of the SAW device in the standardized Raytheon Format.

C) The Raytheon Format, along with some other device specifications, is used as input to program ELECTRO (ELECTRO is similar to program HUGHES). Program ELECTRO produces printed output and a magnetic tape. This magnetic tape contains the instructions that will eventually be used by the Sperry Univac Pattern Generator to produce a glass master.

D) The magnetic tape is used as input to program ELCHECK (ELCHECK is similar to HUESCHK). ELCHECK is a simulation program that simulates the Sperry Univac Pattern Generator. As output it produces a plotted picture of the master.

E) The device engineer looks at the plotted picture of the master to make sure it is what one expects. If the device has been incorrectly described, or the magnetic tape is faulty, the problems will hopefully be detected at this point.

F) The magnetic tape, the printed output from program ELECTRO and the plotted picture from program ELCHECK, are sent to, for example, the Sperry Univac Company. Here, using the magnetic tape (only) as input to the Pattern Generator, they produce a glass master. Note that this magnetic tape is not first processed on an IBM computer as is the case with the tapes for Hughes; and that no printed output is needed to produce the glass master. The magnetic tape contains all the necessary information with minor exceptions such as the dimensions of the glass master and whether the machine will accept inputs in meters or english units. In practice the defaults are almost always used.

G) The Sperry Univac Company compares the glass master they have produced to the plotted picture and to information from the printed output. If they are satisfied they have correctly produced the glass master they send it and all other information back to the device engineers (RADC).

H) The device engineers verify the correctness of the glass master.

I) A direct projection technique is used to produce the final SAW Device as a metal pattern on a piezo-electric crystal.

Program ELECTROXY and program ELCHECKTEK are the other two programs described in this section. Program ELECTROXY is a more versatile version of program ELECTRO. Program ELCHECKTEK is similar to program ELCHECK except it produces a picture on the Tektronix machine instead of on the CALCOMP Plotter.

6.2 The Glass Master

Program HUGHES always produces negative masters. A negative master is described in Section 5.1. On a finished glass master some of the glass is clear and some of the glass is black. When a negative master is produced a photoresist stripping procedure is used so that the clear areas on the glass become metal areas on the piezoelectric crystal, and where the glass is dark there is no metal.

When a positive master is produced a photolithographic chemical-etch procedure is chosen such that the clear areas on the master become clear areas on the piezoelectric crystal and where the master is dark the piezoelectric crystal has metal.

Programs ELECTRO and ELECTROXY are both capable of producing either a positive or negative master. Which kind of master is used depends on several factors. (See Section 6.3.6.4 for a further description of a negative master.)

6.3 Program ELECTRO

6.3.1 Introduction

Program ELECTRO accepts as input the Raytheon format from program CONVERT, and the $ELECTRO namelist. As output it produces printed output, to be used by both the device engineers and, for example, the Sperry Univac Company, and a magnetic tape that directly gives instructions to the Sperry Univac Pattern Generator. The magnetic tape is all the Sperry Univac Company needs to produce the glass master (as described in Section 6.1).

6.3.2 The Pattern Generator

(Read Section 5.2.2 first.) The difference between a Step and Repeat Machine and a Pattern Generator is that with the Step and Repeat Machine the reticle sizes can only be changed by stopping the machine and changing the reticle plate. The proper reticle to use cannot be determined from information on the magnetic tape.

The Pattern Generator works on a different principle. It contains a single plate with a rectangular reticle whose dimensions can vary according to instructions on the magnetic tape. There are some important advantages to this method. Reticles do not have to be cut out in advance. The operator has only to get the process started. After that the whole process is done automatically from start to finish, without human intervention. There is no need for the machine to pause while the operator changes reticles.

The disadvantages to this method are:

A) Only rectangular reticles can be used. Programs ELECTRO and HUGHES only use rectangular reticles, however, applications do exist which call for non-rectangular reticle shapes.

B) When the reticle size is set and reset dynamically there is a greater possibility for loss in accuracy than when fixed reticle sizes are used.

The minimum and maximum reticle dimensions are as follows:

Minimum Reticle Width: 2.54E-8 Meters

Minimum Reticle Length: 2.54E-8 Meters

Maximum Reticle Width: 1.524E-3 Meters

Maximum Reticle Length: 1.524E-3 Meters

6.3.3 Magnetic Tape Format

The format of the tape is as follows:

A) The tape is 9 track, 800 bits per inch, odd parity written in fixed records of 800 bytes. The tape is coded in EBCDIC.

B) The tape must be a "No Label" tape. That is, there must be no introductory material such as title of job, coding instructions, etc. The tape must start immediately with the instruction format described.

C) The first character of each record is the single quote (').

D) Following the last instruction in any record the remaining characters must be periods. There must be at least one period per record. All characters following the first period of a record must be periods.

E) The character "X" followed by from one to seven digits specifies the X position of the center of the reticle. The character "Y" followed by from one to seven digits specifies the "Y" position of the center of the reticle. These digit sequences may represent either units of 10^{-6} inches or 10^{-7} meters. (An operator option that must be consistent with the software used to produce the magenetic tape.) Program ELECTRO is capable of using either units; however, English units have been adopted as standard output (input is always in metric). Thus the command X1000000, for example, would place the reticle one inch to the right of the origin (i.e. a horizontal distance of one inch). The X or Y character sequences described above must never cross record boundaries. Also, they must never end as the 800'th character of a record since there must always be at least one period in every record.

F) The character "W" followed by from one to seven digits specifies the new width for the reticle. The character "V" followed by from one to seven characters specifies the new height of the reticle. One may set a new reticle width without setting a new reticle height (or vice versa). The two commands are independent of each other. The units used in setting the reticle are either English or Metric (see Part E of this section). The W or V character sequences described above must never cross record boundaries. Also, they must never end as the 800'th character of a record since there must always be at least one period in each record.

G) The character "S" (followed by no digits) tells the machine to "Flash". Flashing is described in Section 5.2.2.

H) The character "Z" (followed by no digits) tells the machine to pause. The Z command is used to end the job (the master being complete). There is usually one Z command per tape. (Unless KEARNS = .True., see Section 5.2.6.4 for definition of KEARNS), a Z command must be the last command in a record. All characters in a record following the Z command must be periods.

Note that the tape used by Sperry Univac is 800 characters per record. This is compatible with CDC equipment which also uses tapes with 800 characters per record. Thus the magnetic tape produced by program ELECTRO can be directly sent to Sperry Univac without first being processed on IBM equipment (see Section 5.2.3).

See Figure 6.1 for an example of several records produced by program ELECTRO.

6.3.4 Input to Program ELECTRO

Input to program ELECTRO consists of TAPE9, TAPE28, TAPE29, and the $ELECTRO namelist.

6.3.4.1 TAPE9

(See Section 5.2.4.1)

6.3.4.2 TAPE28

(See Section 5.2.4.2)

6.3.4.3 TAPE29

(See Section 5.2.4.3)

6.3.4.4 Namelist $ELECTRO

Namelist $ELECTRO contains variables that define additional specifications of the glass master. These variables will be discussed in the following sections.

6.3.4.5 ELECTRO Title Card

The ELECTRO title card goes right before the $ELECTRO namelist in the input deck setup. The first four characters of this card may be any alphanumeric character. The fifth and sixth characters must be digits. All other characters may be any valid character. The first six characters are used as the label name on the glass master. (These first six characters are actually flashed onto the glass master.) All other characters are just for user information in the printed output.

```
RECORD NUMBER  15
    *V3000W1600X349301Y53100SY56100SY53100SY50100SY47100SY44100SY41100SX350900Y62100SY59100SY56100SY5310
    0SY50100SY47100SY44100SY41100SY38100SX352500Y62100SY50100SY38100SX354100Y62100SY50100SY38100SX355700
    Y62100SY50100SY38100SX357300Y62100SY50100SY38100SX358900Y52100SY59100SY50100SY47100SY44100SY41100SY3
    8100SX760500Y59100SY47100SY44100SY41100SX373300Y59100SY56100SY53100SY50100SY47100SY44100SY41100SX374
    900Y62100SY59100SY56100SY53100SY50100SY47100SY44100SY41100SY38100SX375500Y62100SY50100SY38100SX37810
    0Y62100SY50100SY38100SX379700Y62100SY50100SY38100SX381300Y62100SY50100SY38100SX382900Y62100SY59100SY
    50100SY47100SY44100SY41100SY38100SX384500Y59100SY47100SY44100SY41100SX397300Y59100SY56100SY53100SY50
    100SY47100SY44100SY41100SX398900Y62100SY59100SY56100SY53100SY50100SY47100SY44100SY41100SY38100......


RECORD NUMBER  17
    *SY400500Y62100SY50100SY38100SX402100Y62100SY50100SY38100SX403700Y62100SY50100SY38100SX405300Y62100S
    Y50100SY38100SX405900Y62100SY59100SY50100SY47100SY44100SY41100SY38100SX408500Y59100SY47100SY44100SY4
    1100SX421300Y59100SY56100SY53100SY50100SY47100SY44100SY41100SX422900Y52100SY59100SY56100SY53100SY501
    00SY47100SY44100SY41100SY38100SX424500Y62100SY50100SY38100SX426100Y62100SY50100SY38100SX427700Y62100
    SY50100SY38100SX429300Y62100SY53100SY38100SX430900Y62100SY59100SY50100SY47100SY44100SY41100SY38100SX
    432500Y59100SY47100SY44100SY41100SX445300Y59100SY56100SY53100SY50100SY47100SY44100SY41100SX446900Y62
    100SY59100SY56100SY53100SY50100SY47100SY44100SY41100SY38100SX448500Y62100SY50100SY38100SX450100Y6210
    0SY50100SY38100SX451700Y62100SY50100SY38100SX453300Y62100SY50100SY38100SX454300Y62100SY59100........


RECORD NUMBER  18
    *SY50100SY47100SY44100SY41100SY38100SX456500Y59100SY47100SY44100SY41100SX469300Y59100SY56100SY53100S
    Y50100SY47100SY44100SY41100SX470900Y62100SY59100SY56100SY53100SY50100SY47100SY44100SY41100SY38100SX4
    72500Y62100SY50100SY38100SX474100Y62100SY50100SY38100SX475700Y62100SY50100SY38100SY477300Y62100SY501
    00SY38100SX478900Y62100SY59100SY50100SY47100SY44100SY41100SY38100SX480500Y59100SY47100SY44100SY41100
    S7..................................  ..............................................................
    ....................................................................................................
    ....................................................................................................
    ....................................................................................................

---MASTER LABEL (TAPE NAME) COMPLETE---


--MASTER 1 COMPLETED---
```

Figure 6.1 Records Written on a Tape Created by Program ELECTRO

(Each full row is 100 characters long)

6.3.5 Output of Program ELECTRO

The output of program ELECTRO consists of TAPE48, TAPE49, and printed output.

6.3.5.1 TAPE49

TAPE49 contains the instructions to the Sperry Univac or other ELECTROMASK Pattern Generator. TAPE49 is a physical tape that is described in Section 6.3.3.

6.3.5.2 TAPE48

TAPE48 contains the same information as TAPE49. The difference is that TAPE49 is a physical tape, whereas TAPE48 is a disk file. TAPE48 is used as input to the Pattern Generator Simulation Program ELCHECKTEK. TAPE48 is needed because a physical tape cannot be read from the Tektronix Machine.

6.3.5.3 Printed Output

The printed output from program ELECTRO contains:

A) A printout of the records on TAPE49 similar to Figure 6.1.

B) A page containing the final values of important variables in program ELECTRO. This page is used by the programmer to help verify the software part of the fabrication process.

C) A page containing a physical description of what the glass master should look like, is useful to the people at the master generation facility and the engineers at RADC to help verify the final product.

6.3.6 Fabrication Specifications Of The Glass Master

6.3.6.1 Introduction

Many of the specifications for fabricating the glass master (usually two inches by two inches), such as the number of fingers per comb, relative overlap of fingers, relative distance of fingers from each other, finger widths, etc., come from the Raytheon format on TAPE9.

Other specifications, such as the actual maximum overlaps of the combs, the number of combs, etc., come from TAPE28, containing the $CONVERT namelist.

The Master Name Label comes from the ELECTRO title card as described in Section 6.3.4.5.

Still other master specifications, such as the placement of the combs on the master, size of the pads, magnification factor, etc., have not yet been discussed.

Some of the remaining specifications are defined according to fixed formulas, others are defined by TAPE9 or the $ELECTRO namelist.

6.3.6.2 The $ELECTRO Namelist Variables And Their Type

PATHCL	real
MASKCL	real
PAD2PAD	real
DIS	real array of length four
PADWID	real
CONOUT	logical
METRIC	logical
OUTLINE	real
FIDWID	real
MASTER	real
KEARNS	logical
SIDEBY	logical
PADONLY	logical
SHORT	logical
POSITIV	logical

6.3.6.3 The $ELECTRO Namelist Variables And Their Defaults

PATHCL	2.54E-2
MASKCL	2.54E-2
PAD2PAD	1.27E-3
DIS	0.,0.,0.,0.
PADWID	2.0E-3
CONOUT	.True.
METRIC	.True.
OUTLINE	1.27E-3

FIDWID	1.27E-3
MASTER	10.0
KEARNS	.False.
SIDEBY	.False.
PADONLY	.False.
SHORT	.False.
POSITIV	.False.

(All units are in Meters)

6.3.6.4 The $ELECTRO Namelist Variables And Their Functions

The functions for the variables PATHCL, MASKCL, PAD2PAD, DIS, PADWID, CONOUT, FIDWID, MASTER, KEARNS, SIDEBY, PADONLY, and SHORT are described in Section 5.2.6.4.

METRIC: If METRIC = .False., then the units on the output magnetic tape are in micro inches, i.e. 1 unit equals 10^{-6} inches.

If METRIC = .True., then the units are in tenths of micrometers, i.e., 1 unit equals 10^{-7} meters.

POSITIV: If POSITIV = .True., a positive master is produced. If POSITIV = .False., a negative master is produced (see definition of "OUTLINE" for a further description of a negative master).

OUTLINE: Suppose one were to produce a positive glass master using program ELECTRO. Now suppose another glass master is produced using all the same programs and input parameters except that "POSITIV" equals .False. (that is, a negative master is produced). One might expect that the negative master would be the exact opposite of the positive master (that is, wherever the positive master is black the negative master is clear, and visa versa). Such is not the case. To produce a true negative master would require a great number of flashes, and it has been found that this is unnecessary.

Consider that fiducial marks are used to line up the master center lines. The technologist does not care if the fiducial marks are formed by two black squares on clear glass, or if they are formed by two clear squares surrounded by a sea of black.

The Master Label Name is used to identify the master. It can be identified just as easily if it is written in black surrounded by clear as it can clear surrounded by black.

Similarly it has been found that it is not necessary to surround the actual combs by black over the whole master. It is only necessary to surround the combs by a black "Outline". The width of this outline is determined by the variable "OUTLINE" (see Figure 6.2).

6.3.6.4.1 Variables RET, BULGE Not Needed

Since the setting of the reticles is software controlled, it is not necessary for the user to specify values for BULGE or RET (from the $HUGHES namelist). All reticle setting is done automatically by the software.

6.3.6.5 Other Specifications Not Directly Defined by the $ELECTRO Namelist or TAPE9.

6.3.6.5.1 Horizontal Centering Of Combs On The Glass Master

(See Section 5.2.6.5.1)

6.3.6.5.2 Verticle Centering of Combs On The Glass Master

(See Section 5.2.6.5.2)

6.3.6.5.3 PAD Length Of The I'th Comb

(See Sections 5.2.6.5.3 and 6.3.6.5.1)

6.3.6.5.4 Centering Of Fingers On Pads

(See Section 5.2.6.5.4)

6.3.6.5.5 Fiducial Marks

(See Section 5.2.6.5.5)

6.3.6.5.6 NMINTH Marks

Program ELECTRO produces no NMINTH marks (see Section 5.2.6.5.6).

6.3.6.5.7 Master Label

(See Section 5.2.6.5.7)

A)

B)

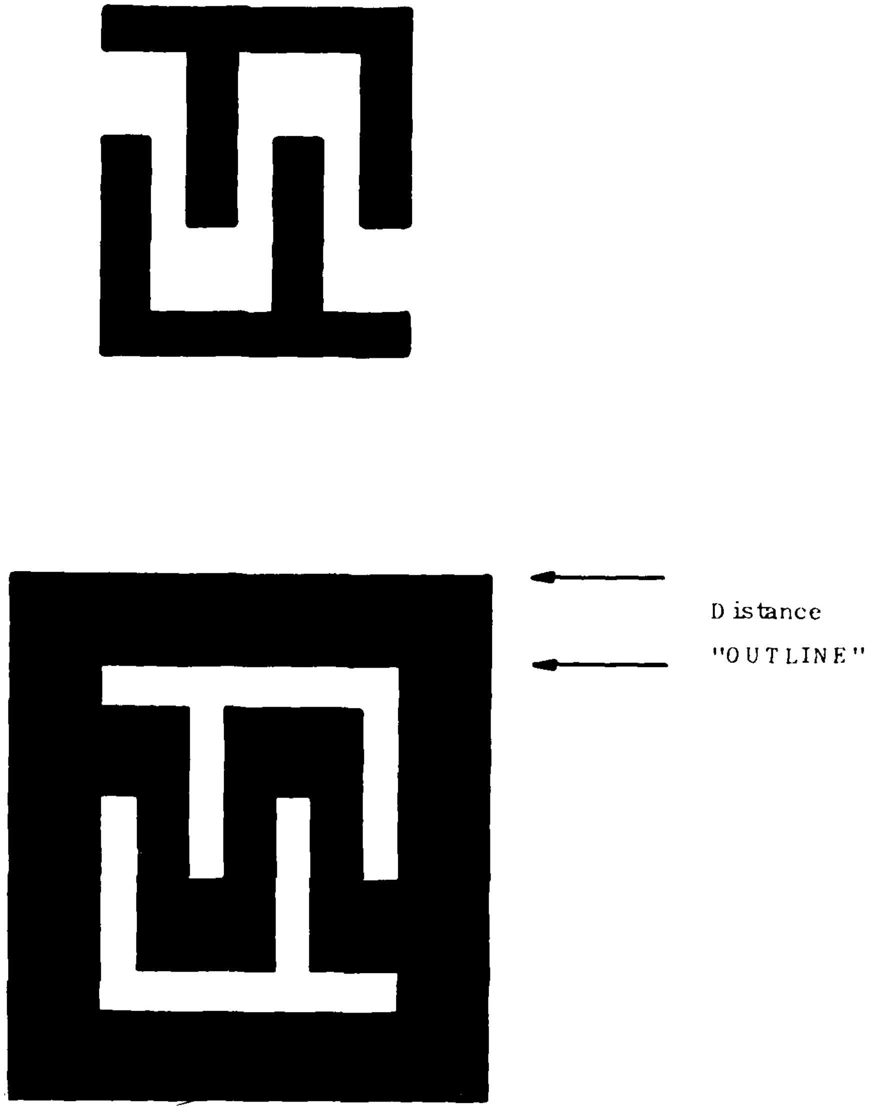


Figure 6.2 <u>Positive and Negative Masters</u>

Diagram A is an example of a comb on a positive master. Diagram B is an example of the same comb on a negative master.

6.4 Program ELECTROXY

6.4.1 Introduction

Program ELECTROXY is basically a more versatile version of program ELECTRO. Program ELECTROXY is very similar to program HUGHESXY. The added features of program ELECTROXY over program ELECTRO are as follows.

A) Program ELECTROXY allows for up to sixteen combs on a master while program ELECTRO allows only a maximum of five combs on a master.

B) Program ELECTRO allows the user only limited flexibility in the placement of the combs on the master while program ELECTROXY allows the user complete flexibility in the placement of the combs on the master.

C) All combs on a master produced by program ELECTRO have the same width pads as each other, this width being determined by the variable PADWID. Program ELECTROXY allows more flexibility in determining pad widths. Both pads on a single comb must still be the same width; however, the pad widths may vary from comb to comb. The pads of a particular comb can even be suppressed entirely if the user so desires.

D) The pad lengths in program ELECTRO are determined by a set formula (see Section 5.2.6.3.3). Program ELECTROXY allows the user almost total flexibility in determining the pad length.

E) All combs on a master produced by program ELECTRO have the same maximum overlap while program ELECTROXY allows varying maximum overlaps from comb to comb.

F) The distance PADS (see Section 3.6.4) is the same for all combs on a master produced by program ELECTRO. Program ELECTROXY allows the variable PADS to vary from comb to comb.

The input to program ELECTROXY consists of TAPE9, TAPE28, TAPE29, the $ELECTRO namelist and the ELECTROXY title card. This is the same as for program ELECTRO, only with program ELECTROXY TAPE9, TAPE28, and TAPE29 must come from program CONVXY, and not program CONVERT, since the formats of these tapes are not exactly the same. (For example, TAPE28 contains the $CONVERT namelist which is not the same for both CONVXY and CONVERT.) Also note that the $ELECTRO namelist is not the same for program ELECTRO as it is for ELECTROXY. The output of program ELECTROXY is the same format as program ELECTRO.

6.4.2 Added or Modified $ELECTRO Namelist Variables Supplying Additional Versatility

PADWID: In program ELECTROXY, PADWID is a real array.

If, for the I'th comb

$$\text{PADWID(I)} \geq 2.54\text{E-}8$$

then the I'th comb will have two pads, each having width PADWID(I).

If PADWID(I) $<$ 2.54E-8

then the I'th comb will have no pads. That is, only the fingers will be fabricated. Note that the variable "NOPAD' (from program HUGHESXY's $HUGHES namelist) does not exist for program ELECTROXY.

Default Value: PADWID(I) = 2.0E-3 for all combs.

PADLEN: PADLEN is an added $ELECTRO namelist variable. PADLEN is a real array which PADLEN allows the user to choose the pad lengths for any comb on the glass master. Both pads on a particular comb, however, must be the same length. The actual length of the pads on the I'th comb of a master is PADLEN(I) * MASTER. As with HUGHESXY this is the <u>only</u> exception to the rule of entering $CONVERT data at final device size and $ELECTRO data at master size. (For example, the PADWID values are absolute and not multiplied by the variable MASTER, i.e., the width of the pads of the I'th comb on the glass master is simply PADWID(I).)

If no value for PADLEN(I) is specifically input by the user (via namelist $ELECTRO) then a temporary default value of PADLEN(I) = .9E+11 is assigned. This enormous value for PADLEN(I) indicates, that although the pad length is not available at the time the $ELECTRO namelist variables are printed out, eventually the pad length will be calculated in the standard way discussed in Section 5.2.6.3.3.

If for a particular comb the user defined value of PADLEN(I) is less than XLEN(I) (see Section 5.2.6.3.3) then PADLEN(I) is set to XLEN(I). That is, PADLEN(I) cannot be less than XLEN(I).

XP: YP: XP and YP are both added $ELECTRO namelist variables: Both XP and YP are real arrays. The value of XP(I) and YP(I) determine the position of the I'th comb. The I'th comb is positoned according to the center of its zero'th gap. (By the "center" of the zero'th gap we mean the exact center both horizontally and vertically.)

Example: XP(3) = 2.54E-2, YP(3) = 2.54E-2. This means the center of the zero'th gap of the third comb will be placed one inch above, and one inch to the right of the origin on the glass master. Default values are seldom used, the user must almost always input these arrays.

Defaults: XP(I) = .002988I

YP(I) = .0254

6.4.3 Other $ELECTRO Namelist Variables in ELECTROXY

CONOUT:	As in program ELECTRO
PATHCL:	No longer used
MASKCL:	No longer used
PAD2PAD:	No longer used
DIS:	No longer used
FIDWID:	As in program ELECTRO
MASTER:	As in program ELECTRO
KEARNS:	KEARNS must equal .False. (default)
SIDEBY:	SIDEBY must equal .False. (default)
PADONLY:	As in program ELECTRO
POSITIV:	As in program ELECTRO
OUTLINE:	As in program ELECTRO
SHORT:	As in program ELECTRO
METRIC:	As in program ELECTRO

6.4.4 Variables From the $CONVERT Namelist Supplying Additional Versatility

OVALAP: The OVALAP values are supplied to program ELECTROXY via TAPE28. If TAPE28 has been created by CONVXY (as it must be to run ELECTROXY) then OVALAP will be a (real) array. The maximum overlap of the fingers of the I'th comb will be MASTER*OVALAP(I) (on the glass master).

PADS: The PADS values are supplied to program ELECTROXY via TAPE28. If TAPE28 has been created by CONVXY (as it must be to run ELECTROXY), then PADS will be a (real) array. (See Section 3.6.4 for a definition of PADS.) The values of PADS will be multiplied by MASTER to create the glass master.

NCOMBS: In program ELECTRO the maximum value for NCOMBS is five, while in program ELECTROXY the maximum value is sixteen.

W1: Unlike program HUGHESXY all the fingers on a master created by ELECTROXY must have the same width. On the glass master this width is W1(1)*MASTER.

6.5 Program ELCHECK

6.5.1 Introduction

Program ELCHECK is a simulation program that simulates the output of the Pattern Generator. Input to the program is TAPE49 from either program ELECTRO or ELECTROXY. Output of the program is a plotted picture of the glass master the Pattern Generator is expected to fabricate.

6.5.2 Input to Program ELCHECK

6.5.2.1 TAPE49

TAPE49 is a physical tape which is created by either program ELECTRO or program ELECTROXY.

6.5.2.2 ELCHECK Title Card (Optional)

The card input to program ELCHECK consists of the ELCHECK title card followed by the $ELCHECK namelist. The first 30 characters of the ELCHECK title card are printed on the output plots and must contain the name and telephone number of the user.

6.5.2.3 The $ELCHECK Namelist

6.5.2.3.1 The $ELCHECK Namelist Variables And Their Type

PLTSIZE:	Real
MSTSIZE:	Real
MAG	Real
METRIC	Logical

6.5.2.3.2 The $ELCHECK Namelist Variables And Their Defaults

PLTSIZE	=	10.
MSTSIZE	=	2.
MAG	=	8.
METRIC	=	.True.

6.5.2.3.3 The $ELCHECK Variables And Their Functions

PLTSIZE: The size of the plotted picture of the glass master is PLTSIZE inches long by PLTSIZE inches wide. (It is assumed that the glass master is square.)

MSTSIZE: MSTSIZE is the length and width (in inches) of the actual master to be fabricated. (It is assumed that the glass master is square.)

MAG: Figure 6.3 is an example of a plotted picture produced by program ELCHECK. Note the rectangle outlined by the dotted line. This portion of the picture will be magnified in the next plotted frame. The magnification will be such that the width of the rectangle will take up the whole width of the paper (that is, the width will be "PLTSIZE", which is usually ten inches). The bottom edge of the rectangle always goes from fiducial mark to fiducial mark. The height of the rectangle will be:

$$\text{PLTSIZE}/(2*\text{MAG}) \quad \text{(inches)}$$

In the example shown in Figure 6.3 MAG = 8. Thus, the height of the rectangle is PLTSIZE/16, and the rectangle will be magnified sixteen times in the next frame. In general the magnification factor will be (2*MAG). If MAG = 0.0, then the magnification frame is suppressed.

METRIC: The value of METRIC in the $ELCHECK namelist should always be the same value as METRIC in the $ELECTRO namelist (see Section 6.3.6.4).

6.5.3 Output of Program ELCHECK

The output of program ELCHECK is the plotted picture of the glass master and printed output giving the values of the $ELCHECK variables.

6.5.4 Differences Between Programs ELCHECK And HUESCHK

Program HUESCHK simulates an Electromask Step and Repeat machine. Program ELCHECK simulates an Electromask Pattern Generator. With a Step and Repeat machine the operator must be given written instructions describing what size reticles to use and when to change them. TAPE50, created by HUGHES or HUGHESXY and used as input by HUESCHK represents this written information. With a Pattern Generator the reticle sizes are given directly by the magnetic tape. Thus, no reticle information is needed by the operators. Similarly TAPE50 is not needed by program ELCHECK. Thus, TAPE50 is neither created by program ELECTRO (or ELECTROXY) nor read by program ELCHECK.

Program HUESCHK allows the user to vary plotting time with the variables PDEN, FDEN, HALF and KPIK. Program ELCHECK does not have this capability.

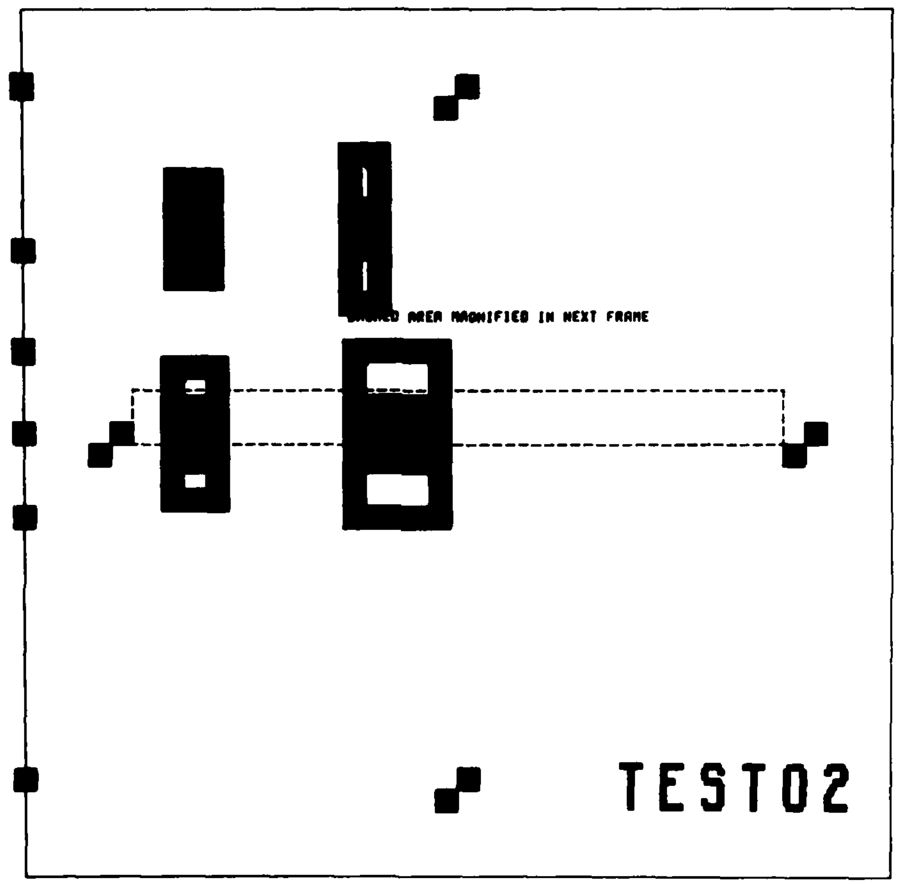


Figure 6.3 Picture of a Negative Master from a Tape Created By ELECTROXY and Simulated by Program ELCHECK

Note that the upper left comb has no Pads. Everything except the dashed line and the words "dashed area magnified in next frame" will be flashed on the glass master.

6.6 Program ELCHECKTEK

6.6.1 Introduction

Program ELCHECKTEK is similar to program ELCHECK. The difference is that program ELCHECK produces the picture of the master on a CAL-COMP plotter. Program ELCHECKTEK displays the picture of the master on the cathode ray tube of the Tektronix machine. By pushing a button on the Tektronix machine the user may obtain a hard copy of the picture shown on the screen. The advantage of using the Tektronix machine is that the turn-around time is immediate. When using program ELCHECK the user must wait to receive the plot. The user may get a picture on the Tektronix machine by using program ELCHECKTEK immediately after program ELECTRO (or ELECTROXY) has been run.

The disadvantages of using ELCHECKTEK are:

A) The quality of the picture (size and resolution) is much poorer on a hard copy produced by ELCHECKTEK than on a plot produced by program ELCHECK.

B) The magnified frame often comes out distorted on the Tektronix machine due to its limited screen size.

It is possible to run program ELECTRO (or ELECTROXY) and ELCHECK in the same run and then go over to the Tektronix machine and run program ELCHECKTEK. In this way the user can know immediately if the master being simulated looks reasonable. If the Tektronix picture is not what the user expected one can go to the computer center and abort the plot job thereby saving valuable plot time. If the Tektronix picture seems reasonable one can wait for the CAL-COMP plot for further details. Program ELCHECKTEK is particularly useful in debugging a new version of program ELECTRO or ELECTROXY. In order to use the Tektronix machine it is advised that the user know something about the Hanscom AFB CDC 6600 time sharing system, otherwise one may run into situations one does not know how to handle (time limits, typing mistakes, etc.).

6.6.2 Input to Program ELCHECKTEK

Input to program ELCHECKTEK consists of TAPE49 and input typed directly into the Tektronix machine.

6.6.2.1 TAPE49

Let us describe where TAPE49 comes from. First consider program ELECTRO (or ELECTROXY). Program ELECTRO (or ELECTROXY) produces both TAPE48 and TAPE49. TAPE49 is a physical tape that cannot be read from the Tektronix machine.

For this reason TAPE48 has been created. TAPE48 contains the same information as TAPE49, only it is a disk file instead of a physical tape. Thus, if the user is planning to run program ELCHECKTEK from the Tektronix machine, TAPE48 must be saved as a permanent file.

Now consider it from the point of view of program ELCHECKTEK. ELCHECKTEK requires a disk file called TAPE49 as input. Thus, the user must create a TAPE49 from the information saved as TAPE48. The details for doing this are described in Sections 6.5.3 and 6.5.4.

6.6.2.2 User Input Directly Into The Tektronix Machine

This will be discussed in Section 6.6.4.

6.6.3 Preliminary Steps Needed To Run Program ELCHECKTEK

In the control card setup, before Program ELECTRO (or ELECTROXY) is run, request TAPE48 as a permanent file. After program ELECTRO (or ELECTROXY) has been run save TAPE48 as a permanent file. An example of this section of the control cards follows:

```
REQUEST,TAPE48,*PF.

ELECTRO.

CATALOG,TAPE48,SAVETAPE48,ID=ELTERMAN,MR=1.
```

6.6.4 Running Program ELCHECKTEK On The Tektronix Machine

Assuming the preliminary steps discussed in Section 6.6.3 have been followed, complete the following dialog with the machine. In the following example underlined text represents text written on the Tektronix machine by the computer. Non-underlined text represents commands typed into the Tektronix machine by the user. Text in parenthesis represents the author's comments. ↓ means the user should press the carriage return on the Tektronix machine. "Δ" is the notation used to mean the user should press the space bar.

LOGIN,NAME,CODE,861444,SUP ↓
(The user must have one's own LOGIN name and code.)

<u>COMMAND</u> SCREEN,80,66.↓

<u>COMMAND</u> ATTACH,ELCHECK,ELCHECKTEK,ID=ELTERMAN,MR=1. ↓

(ELCHECKTEK must be cataloged under user's ID as a binary file.)

<u>COMMAND</u> ATTACH,TAPE49,SAVETAPE48,ID=ELTERMAN,MR=1.↓

<u>COMMAND</u> ATTACH,TEK,TEKLIB.↓

COMMAND LIBRARY,TEK. ↓

COMMAND ELCHECK. ↓

(Wait for the computer to print out a question mark)

?

△$ELCHECK$ ↓(options may be input if the user wishes)

(Wait for the computer to write the following message.)

1 GO/ERASE
2 GO/SAVE
3 EXIT
ENTER OPTION

(The user should respond with)

1↓

(The above message may be repeated several times, each time the user should type)

1↓

(until a picture similar to Figure 6.10 comes on the screen. At this point the user can study the picture and get a hard copy if one wishes by pressing the "Copy" button. (Make sure the hard copy machine is turned on and warmed up. After leaving, turn the hard copy machine off.) The user may type)

1↓

(Once more to see the (probably distorted) magnified frame. After this type)

3↓

COMMAND LOGOUT

(Push the screen reset button and you are now ready to leave.)

7.0 PROGRAM TRPLOT

7.1 General Description

Program TRPLOT plots a picture of a transducer directly from the output (i.e. TAPE9 and TAPE29) of program CONVERT. Since there are no namelists, nor other input options, program TRPLOT is very easy to use. If program CONVERT has been run correctly then program TRPLOT can be run with no further preparation (see Figure 7.1).

Program TRPLOT does not plot a scaled picture of the transducer but rather plots the transducer in a way that makes it easy to see the relative overlap of any two elements.

The distance Ho[0] is always plotted eight inches long and all other Ho[N] or Ho[NTAP] values are scaled accordingly.

Example: If Ho[0] = 1
and Ho[1] = .8

Then the plotted overlap at the 0'th gap (or tap) will be 8 inches long, while the plotted overlap at the 1'th gap (or tap) will be 6.4 inches. The distance EXTRA (see Section 3.6 under "PADS") is also scaled. Thus, the plotted distance of EXTRA is (8)(EXTRA). The width of the bus bars ,however, are always 1/2 inch.

From the above description we can see that the vertical dimensions (except for the bus bars) of the plot are scaled to that of the actual transducer.

In the horizontal direction the dimensions of the plot are also in scale with the actual transducer. However, the scale factors for the horizontal and vertical dimensions are not necessarily the same.

Example: An actual transducer has an overlap of 500E-6 meters at the zero'th gap. (This is the default value of OVALAP, from the $CONVERT namelist). The actual widths of the electrodes are 1.2E-6. (This is the default value of both W1 and W2, from the $CONVERT namelist.)

In the vertical direction the length of the overlap at the zero'th gap will be increased from 500E-6 meters to 8 inches.

Eight inches equals .2032 meters.

$$\frac{.2032 \text{ Meters}}{500E\text{-}6 \text{ Meters}} = \text{A magnification factor of 406.4 in the vertical direction}$$

In the horizontal direction the width of the electrodes will be increased from 1.2E-6 meters to 1/30 of an inch.

1/30 inch = .847E-3 meters

$\frac{.847E-3 \text{ Meters}}{1.2E-6 \text{ Meters}}$ = A magnification factor of 705.6 in the horizontal direction.

Thus, the overall length of the plotted transducer will be about 705 times the length of the actual transducer, while the width of the plotted transducer will only be about 406 times the width of the actual transducer.

The result is simply that, a plot from program TRPLOT is easy to look at and gives one a visual description of some of the transducer's important properties, but is not an accurate picture of the real transducer.

On the left and right sides of the plotted transducer are plotted two axes (the Y-axes). The total length of the Y-axes is ten inches. There are tic marks every inch along the axes. The distance Ho[0] is plotted 8 inches long, but the value of Ho[0] is 1. The Y-axes are scaled according to this. The bottom of the zero'th overlap is Y = 0 and the top of the zero'th overlap is Y = 1.

On the top and bottom of the plotted transducer are plotted two axes (the X-axes). The X = 0 point is right below the left edge of the leftmost finger. Tic marks occur every inch and the axes are scaled such that the numbers represent the actual length of the physical transducer in millimeters. (In Figure 7.1 the real length of the transducer would be about .05 millimeters from first finger to last finger.)

7.2 Input

Input to program TRPLOT consists of TAPE9 and TAPE29 (from program CONVERT) and an optional plot identification card.

TAPE9 contains Raytheon descriptions of up to five separate transducers, each separated by an end of file mark. Program TRPLOT will produce a plot for each transducer on TAPE9.

TAPE29 contains title card information from the title cards used in the Ho[N] and ACT[M] (or Ho[NTAP] and TCA NTAP) decks, along with the values of EXTRA and DUMMY (DUMMY is from the $CONVERT namelist (see Section 3.6) for each transducer.)

The optional plot identification card is used to provide the name of the initiator. The first thirty characters are read from the card and printed on the plot. If this card is omitted, then the

identification default is: "SLOBODNIK X3716 TRANSDUCER PLOT". Thus, the input deck for program TRPLOT is either a label card followed by a 789 card, or just a 789 card.

7.3 Output

Output from program TRPLOT consists of the plot, printed output and TAPE13.

The printed output consists solely of the number of the transducer on TAPE9 (i.e. 1, 2, 3, 4 or 5) followed by the title(s) from the corresponding Ho[N] and/or ACT[M] decks (or Ho[NTAP] and/or TCA [NTAP] decks).

TAPE13 contains the coordinates of the plotted transducers, which can be dumped out as printed output if the user so desires. TAPE13 has no other purpose.

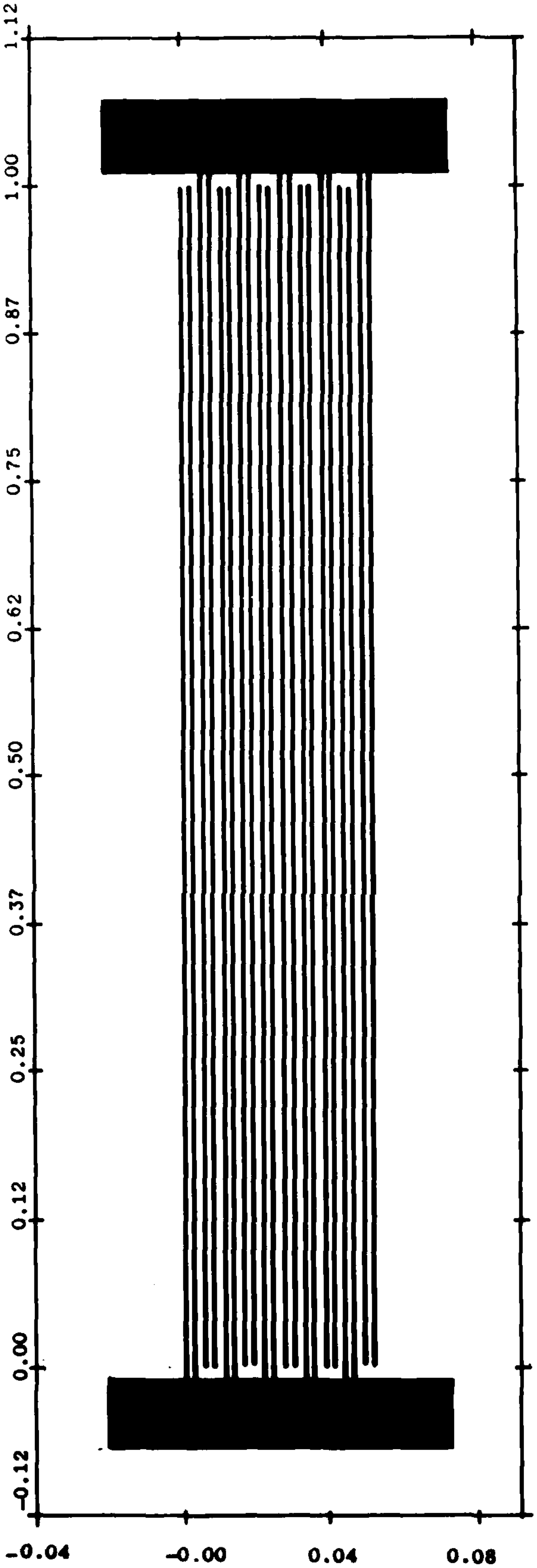


Figure 7.1 Plot from Program TRPLOT

8.0 STANDARD SAW DEVICE ANALYSIS PROGRAMS

8.1 Introduction

The programs in this section are used to mathematically analyze a SAW device. Three of the programs discussed in this section (GRAF, GRAFTEK, SEEIT) were designed and programmed by ACSI personnel. These programs will be discussed in detail. Four of the programs discussed in this section (COMBS, MATCH3, PULSE, TRANS) were not written by ACSI personnel. Hence only program modifications implemented by ACSI personnel and the manner in which these programs interface with ACSI programs will be discussed.

8.2 Program COMBS

8.2.1 ACSI Modifications

A) The real array X was originally dimensioned to 4000. It is now dimensioned to 9000.

B) The integer IDIM was originally set to 4000 in a data statement. It is now set to 9000 (in the same data statement).

8.2.2 Interfacing With ACSI Programs

Program COMBS accepts as input (among other inputs) TAPE9 created by program CONVERT. (TAPE9 from program CONVXY should not be used.) As output program COMBS produces TAPE10 that can be used by MATCH3 or by program GILPM (GILPM is an ACSI program which will be discussed in Chapter 9).

8.3 Program MATCH3

8.3.1 ACSI Modifications

Two subroutines were added to MATCH3. Subroutine SMITH and subroutine CHART. Together these two routines produce a SMITH CHART plot of the data

8.3.2 Interfacing With ACSI Programs

MATCH3 accepts TAPE10 (from program COMBS) as input. As output it produces SMITH CHARTS, TAPE11, and TAPE22. TAPE22 contains insertion loss versus frequency data and is used by program GRAF, GRAFTEK and GILPM.

8.4 Program TRANS

8.4.1 ACSI Modifications

A number of variables were added to the $TRANS namelist. If LINEAR = .True., then a linear plot is produced and the following variables are envoked:.

Variable (with defaults)

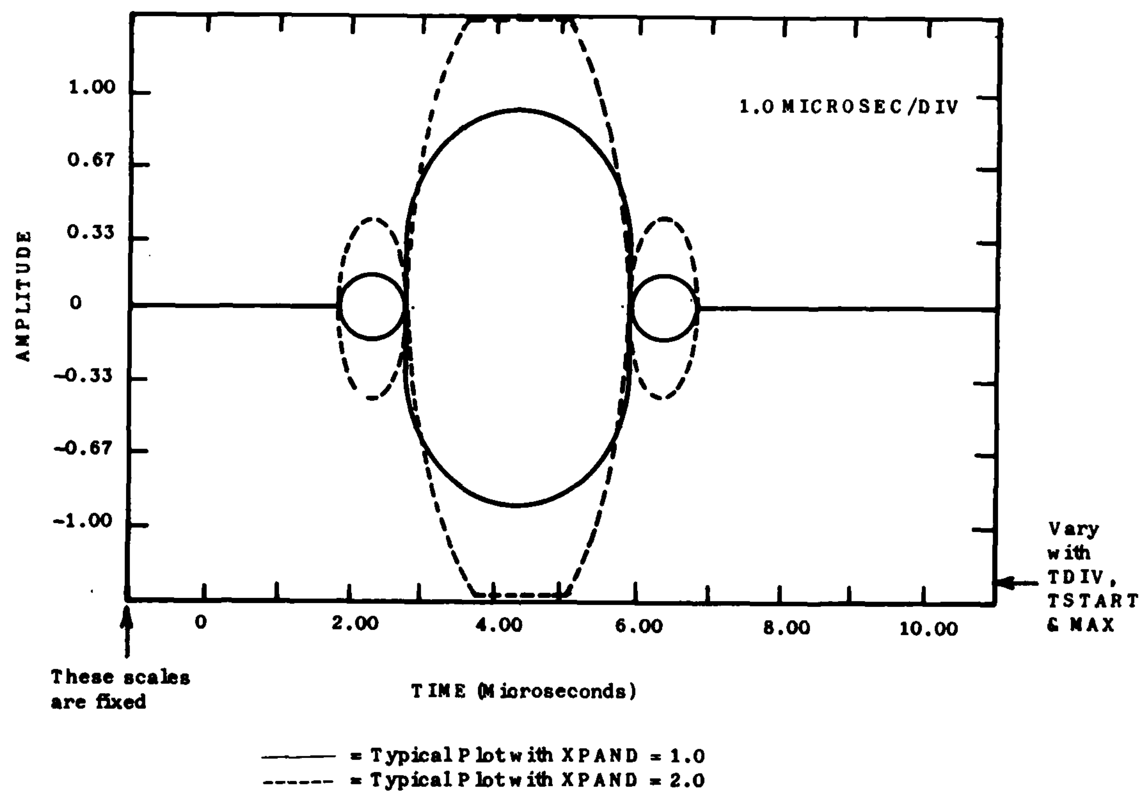
TDIV = 1.0, 1.0, 1.0 (number of microseconds per divison on horizontal scale)

TSTART = 0.0, 0.0, 0.0 (value, in microseconds, of the second tic mark on horizontal scale)

MAX = F, F, F (a logical array which, when .True., causes the maximum amplitude of the curve to be centered on the plot thus overriding TSTART)

XPAND = 1.0, 1.0, 1.0, (the maximum point on the plot is normalized to XPAND).

In addition to the above, plots are to be on a linear scale and labels are to be large-size (similar to PLTDAT plots). Note how plot made is symmetric in positive and negative amplitude.



If LINEAR = .False, then a db plot is produced.

The variable XPAND is used to normalize the curve. The maximum point on the plot is normalized to "XPAND".

8.4.2 Interfacing With ACSI Programs

TRANS does not directly interface with any ACSI program. Indirectly it accepts input through CONVERT in the following sequence.

CONVERT → COMBS → MATCH → TRANS.

8.5 Program PULSE

8.5.1 ACSI Modifications

None.

8.5.2 Interfacing with ACSI Programs

PULSE does not directly interface with any ACSI programs, but provides input to program TRANS. TRANS indirectly interfaces with program CONVERT.

8.6 Program GRAF

8.6.1 Introduction

Program GRAF plots insertion loss in decibels as a function of frequency in megahertz. Data for program GRAF comes from Tape22 which is produced by program MATCH2 or GILPM. There is only one set of data on TAPE22; however, this one set of data may produce as many as four different graphs. Each graph, although using the same data, may look different because of different scaling factors and/or normalization techniques. Namelist $PLTDAT contains the variables that determine both the number of graphs to be produced and the scaling and normalization procedures for each graph produced. Each graph (independent of $PLTDAT variables) has an X-axis ten inches long and a Y-axis eight inches long. Both axes have one tic mark every inch. The bottom of the Y-axis has a value of 80 decibels. All other values assigned to tic marks are controlled by the variables on the $PLTDAT namelist.

8.6.2 TAPE22 (from program MATCH2 or program GILPM)

The first record on TAPE22 contains a heading (to be plotted on each graph) 100 characters long followed by a logic field of length 1. The actual statement used to read this record is: READ(22,100) HEAD, LABLE.

HEAD is an array variable of length 10 and LABLE is a logical variable. The format used to read this record is:

100 FORMAT(10A10, L1)

If LABLE = TRUE then the second record is a second heading also 100 characters long (to be plotted on each graph). The format of the second heading is (10A10).

The next record (either the second or third record) contains the variables ICOUNT, DV, SIZE, PROGID, NO. The format of this record is (I5, 2F10.5, 3A10, I2). The variable functions are as follows:

ICOUNT: The number of records to follow

DV: Presently not used.

SIZE: The value of size from TAPE22 is presently not used. The value of size is reset to ten shortly after being read in. Size is the length of the X-axis for each graph.

PROGID: This is the plot identification to appear at the beginning of the plot (usually containing the name and telephone extension of the program user). PROGID is initially read in from the card preceding the $MATCH namelist.

NO: A number originally defined by the $MATCH namelist, NO is simply a number used for identification purposes. This number gets plotted 9 1/4 inches above the plotting origin (the plotting origin is in the lower left corner of the graph).

The next "ICOUNT" records contain the insertion loss data. The format of these records is (2020). These records are read into the variables FF(I) and XLOSS(I) for I = 1 to ICOUNT. The array FF contains the frequency data in Hertz (note that the graph itself has units of megahertz along the X-axis). The array XLOSS contains the corresponding insertion loss for each frequency in decibels.

The program user would usually not be concerned with the format of TAPE22. If program MATCH2 is run correctly then TAPE22 will be there when needed by program GRAF.

8.6.3 Namelist $PLTDAT

8.6.3.1 Namelist $PLTDAT Variables And Their Types

DIV: Real array of length 4

FZERO: Real array of length 4

DY:	Real
UNCAL:	Real array of length 4
TONES:	Real array of length 4.

8.6.3.2 Namelist $PLTDAT Variables And Their Defaults

DIV:	0.0, 0.0, 0.0, 0.0
FZERO:	500.E6, 500.E6, 500.E6, 500.E6
DY:	-10.0
UNCAL:	99.0, 99.0, 99.0, 99.0
TONES:	0.0, 0.0, 0.0, 0.0

8.6.3.3 Namelist $PLTDAT Variables And Their Functions

DIV: DIV controls both the units per tic mark along the X-axis and the number of graphs that will be plotted.

The value DIV(I) is the change in frequency from one tic mark to the next (along the X-axis). The value DIV(I) is in Hertz. The units used on the actual plot is maga-hertz. Thus, if one wanted the distance between tic marks to represent an increase in frequency of two mega-hertz, then the appropriate DIV(I) would equal two million.

The program plots as many graphs as there are consecutive positive values of DIV. Examples:

A) DIV(1) = 0 No graphs are plotted

B) DIV(1) = 1.E6 1 graph plotted
DIV(2) = 0

C) DIV(1) = 1.E6 2 graphs plotted
DIV(2) = 2.E6
DIV(3) = 0

D) DIV(1) = 1.E6 3 graphs plotted
DIV(2) = 2.E6
DIV(3) = 3.E6
DIV(4) = -1.E6

E) DIV(1) = 1.E6 4 graphs plotted
DIV(2) = 1.E6
DIV(3) = 1.E6
DIV(4) = 1.E6

FZERO: FZERO(I) is the value of the frequency in the middle (i.e. five inches of the axis is to the left and five inches of the axis is to the right) of the X-axis, of the I'th graph. FZERO(I) is in hertz. The actual numbers plotted on the graph are in megahertz. Thus, if one wanted a value of 50 megahertz at the middle of the I'th graph, then FZERO(I) would equal 50.E6 (i.e. fifty million). Together FZERO(I) and DIV(I) determine the range along the Y-axis.

Example:

If DIV(1) = 1.E6
and FZERO(1) = 50.E6
then the X-axis would range from 45 megahertz to 55 megahertz (see Figure 8.1).

DY: DY controls the values of the tic marks along the Y-axis. As stated in Section 8.6.1 the Y value at the bottom of the graph is 80 decibels. To get the value of the next highest tic mark simply add the value DY. If DY = -10 (default), then the Y-axis will range from 80 to 0 decibels and the values of the tic marks along the Y-axis will be 70, 60, 50, 40, 30, 20 and 10 (see Figure 8.2).

UNCAL: UNCAL(I) is used to shift the data up or down along the Y-axis. That is, a constant is either added to or subtracted from all the numbers in the XLOSS array. The corresponding X values (in the FF array) are not changed (see Section 8.6.2 for definitions of XLOSS and FF arrays).

The rules for UNCAL(I) are as follows:

A) If UNCAL(I) = 99. (Default) then the Y values of the I'th graph are not shifted. That is, array XLOSS remains unchanged.

B) If UNCAL(I) ≠ 99.

Then the smallest value, XLOSSMIN, of the XLOSS array is found and the following constant is computed:

SHIFT = UNCAL(I) - XLOSSMIN

then XLOSS(J) = XLOSS(J)+SHIFT For J=1,
ICOUNT is the length of the XLOSS array (see Section 8.6.2).

Example 1: ICOUNT = 5
UNCAL = 10
XLOSS = 50, 40, 30, 40, 50
SHIFT = UNCAL - XLOSSMIN = 10 - 30 = - 20
and XLOSS(J) = XLOSS(J) - 20 for J=1, ICOUNT
Thus new XLOSS array = 30, 20, 10, 20, 30

Note that the minimum value of the new XLOSS array is equal to UNCAL. This is always the case.

In the actual computer program the XLOSS array is unchanged and the scratch work is done in array XDIFF (XDIFF is the array that actually gets plotted). Logically, however, what happens is as described above.

TONES: If TONES(I) = 0.0 (Default) then the I'th graph is plotted without little "arrows".

If TONES(I) 0.0, then little arrows are plotted on the I'th graph. These arrows are 1 inch long and point directly downward. The top tip of the arrow is three inches above the X-axis. The bottom tip of the arrow is two inches above the X-axis. One arrow always points directly at the X = FZero point. The spacing between arrows is TONES and is in units of Hertz. The units along the X-axis are in megahertz. Thus, if for example, DIV(I) = 1.E6 and TONES(I) = 1.3E6 then the arrows will be spaced 1.3 inches apart. In general the spacing of the arrows will be:

$$\frac{\text{TONES(I)}}{\text{DIV(I)}} \text{ inches}$$

(See Figure 8.3)

8.6.4 Miscellaneous Points of Interest

8.6.4.1 The maximum number of points that can be plotted (i.e. maximum length of both XLOSS and FF arrays) is 2058.

8.6.4.2 If a Y point (i.e. a value in the XLOSS array) falls above or below the limits of the graph (as determined by variable DY), then this value is set to either the top or bottom limit (according to which limit was exceeded) of the graph.

Example: ICOUNT = 7
DY = -5
XLOSS = 90, 80, 50, 30, 50, 60, 70

Thus, the Y-axis goes from 80 at the bottom to 40 at the top. The XLOSS array will be changed to:

XLOSS = 80, 80, 50, 40, 50, 60, 70

In the event that UNCAL $\neq$ 0, then the UNCAL algorithm will be applied to the data before the limits of the data are checked.

8.6.4.3 If an X Point (i.e. a value in the FF array) is to the left or right of the graph limits along the X-axis (as determined by FZERO and DIV), then the point does not get plotted.

```
Example 1:  ICOUNT  =  7
            DIV     =  1.E6
            FZERO   =  50.E6
            FF      =  47.E6, 48.E6, 49.E6, 50.E6, 51.E6,
                       52.E6, 53.E6
```

Hence, for Example 1, the left and right limits of the graph will be X = 45.E6 and X = 55.E6, and thus all the points will be plotted.

```
Example 2:  ICOUNT  =  7
            DIV     =  .5E6
            FZERO   =  52.E6
            FF      =  47.E6, 48.E6, 49.E6, 50.E6, 51.E6,
                       52.E6, 53.E6
```

Then the left and right limits of the graph will be X = 49.5E6 and X = 54.5E6, and thus only the last four points will be plotted.

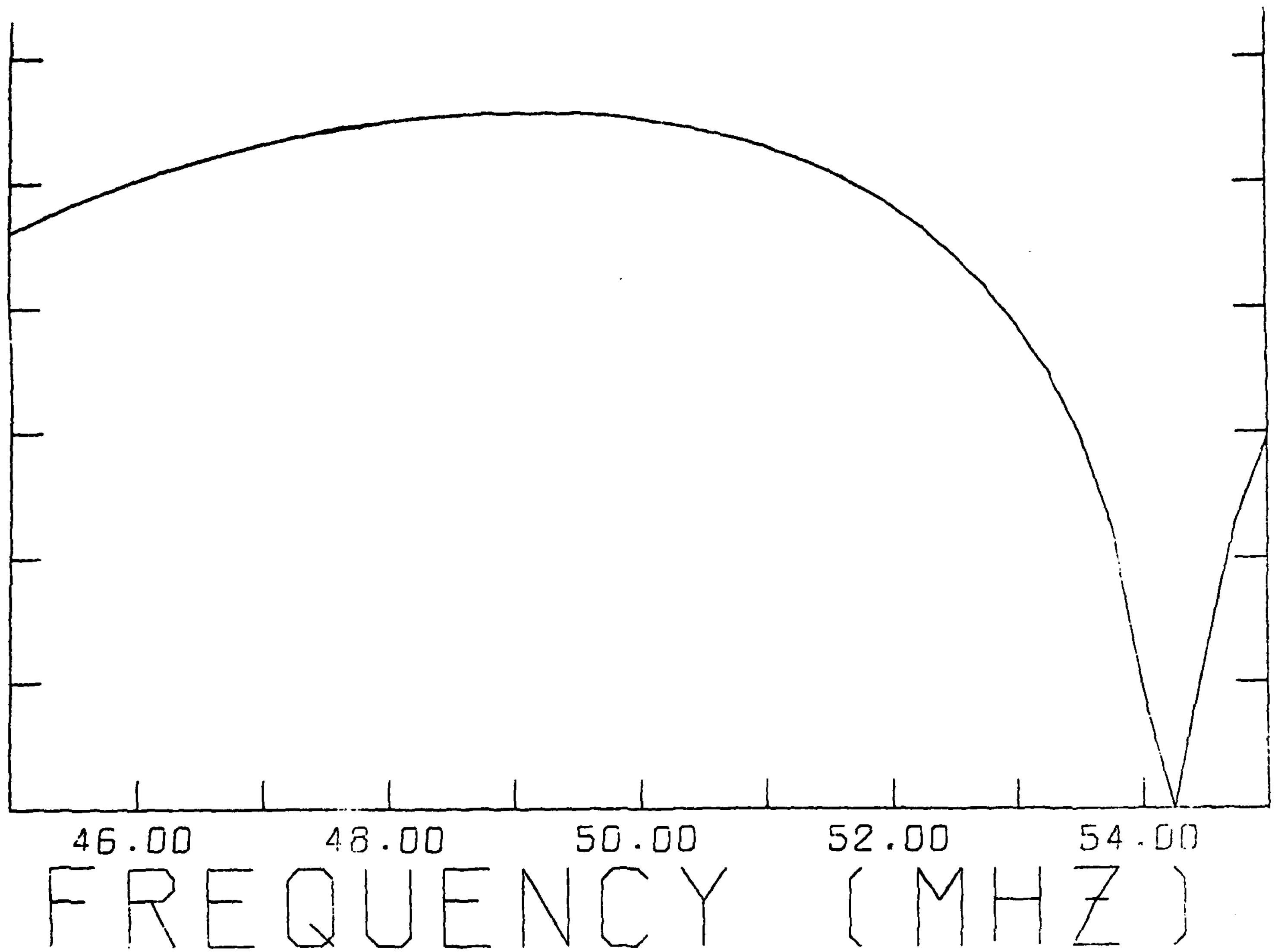


Figure 8.1 Illustration of X-Axis of Graph from Program GRAF

In this case, FZERO=50.E6, DIV=1.E6

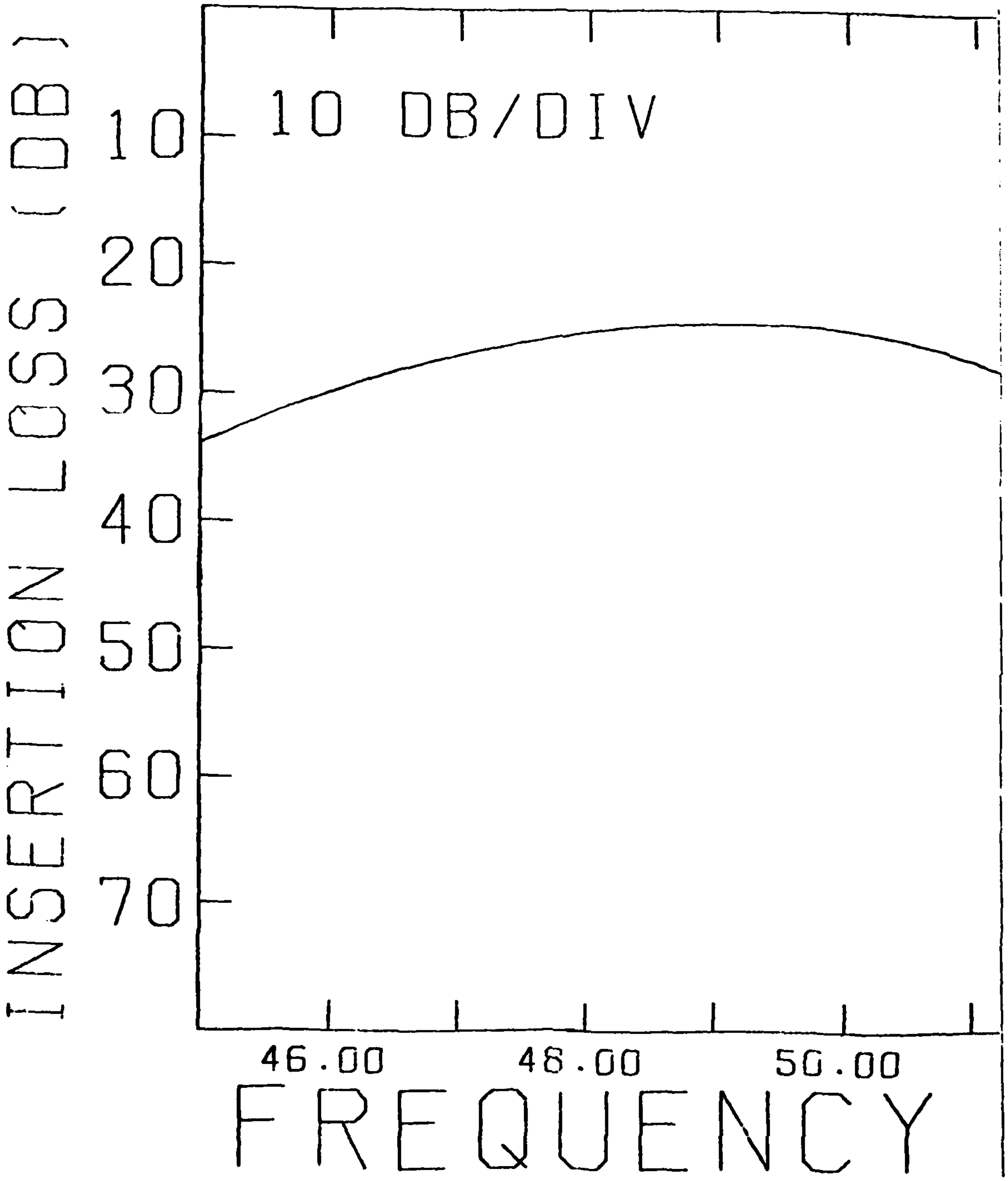


Figure 8.2 Illustration of Y-Axis of Graph from Program GRAF

DY = -10

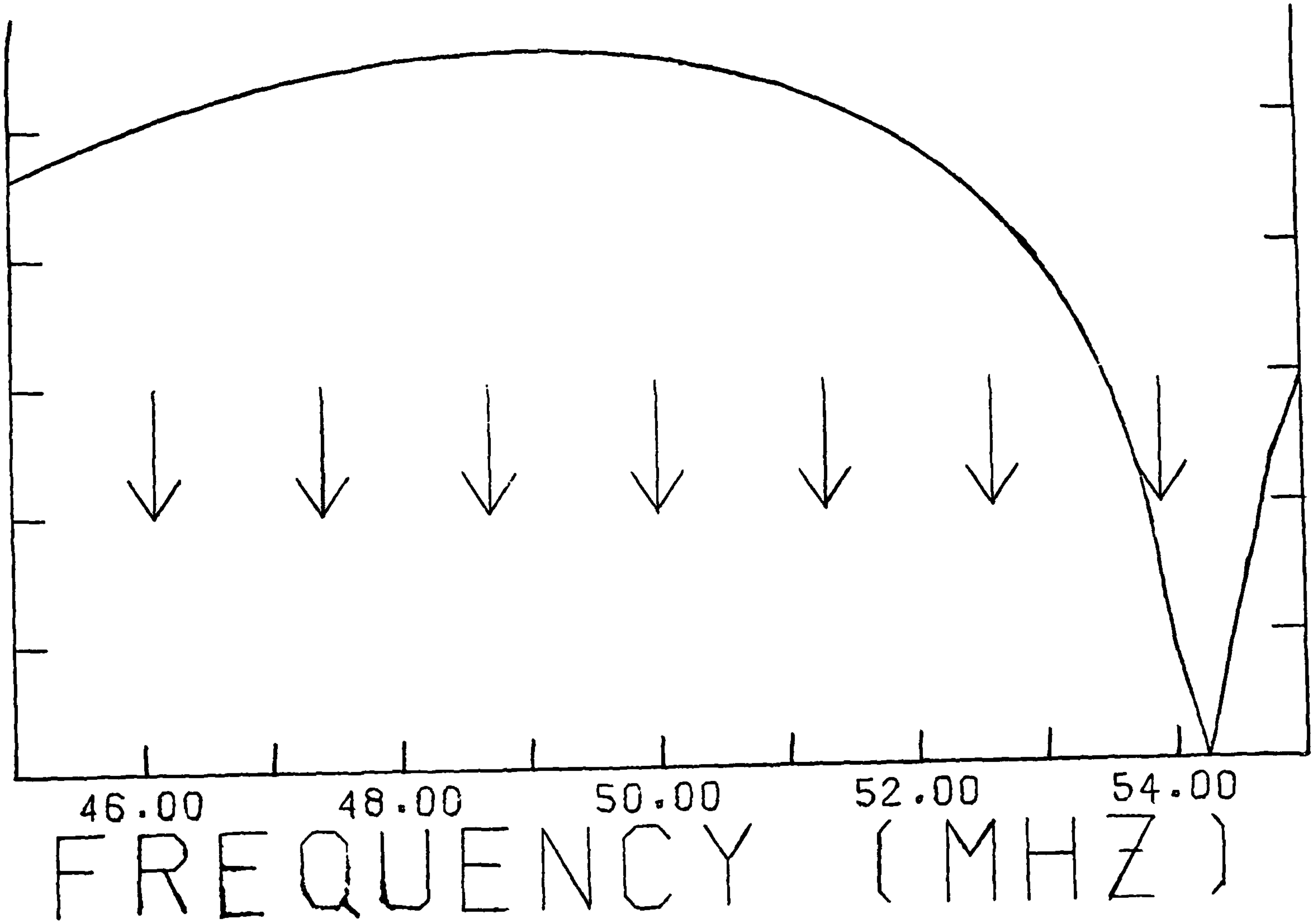


Figure 8.3 GRAF Plot With Tones Option

DIV=1.E6, FZERO=50.E6, TONES=1.3E6

8.7 Program GRAFTEK

8.7.1 Introduction

Program GRAFTEK is similar to program GRAF. The difference is, that with program GRAF the insertion loss plots are plotted on a Cal-Comp plotter while in program GRAFTEK the insertion loss plots are shown on the cathode ray tube of the Tektronix machine. By pushing a button on the Tektronix machine the user may obtain a hard copy of the picture shown on the screen. The advantage of using the Tektronix machine is that the turn-around time is immediate. When using program GRAF the user must wait to receive the plot. The user may get a picture on the Tektronix machine by using program GRAFTEK immediately after program MATCH or program GILPM has been run.

8.7.2 Input to Program GRAFTEK

Input to program GRAFTEK consists of TAPE22 and input typed directly into the Tektronix machine. TAPE22 contains a heading followed by frequency vs. insertion loss data. TAPE22 is usually a disk file created by either MATCH3 or GILPM.

8.7.3 Preliminary Steps Needed To Run Program GRAFTEK

In the control card setup, before program MATCH3 (or GILPM) is run, request TAPE22 as a permanent file. After program MATCH3 (or GILPM) has been run save TAPE22 as a permanent file. An example of this section of the control cards follow:

```
REQUEST,TAPE22,*PF

MATCH3

CATALOG,TAPE22,SAVETAPE22,ID=ELTERMAN,MR=1.
```

8.7.4 Running Program GRAFTEK On The Tektronix Machine

Assuming the preliminary steps discussed in Section 8.7.3 have been taken, complete the following dialog with the Tektronix machine. In the following example underlined text represents text written on the Tektronix machine by the computer. Non-underlined text represents commands typed into the Tektronix machine by the user. Text in parenthesis represents the author's comments. ↓ means the user should press the carriage return on the Tektronix machine. "Λ" is the notation used to mean the user should press the space bar.

LOGIN,NAME,CODE,8614444,SUP
(The user must have one's own LOGIN name and code)

<u>COMMAND</u>   SCREEN,80,66. ↓

COMMAND   ATTACH,GRAF,GRAFTEK,ID=ELTERMAN,MR=1. ↓

(GRAFTEK must be cataloged under user's ID as a binary file)

COMMAND ATTACH,TAPE22,SAVETAPE22,ID=ELTERMAN,MR=1. ↓

COMMAND ATTACH,TEK,TEKLIB. ↓

COMMAND LIBRARY,TEK. ↓

COMMAND GRAF.

(Wait for the computer to print out a question mark)

?

Δ$PLTDAT FZERO=500.E6,300.E6, DIV=10.E6,20.E6,UNCAL(2)=30$

(The above input parameter list is an example; the user may choose any values (consistent with the data). In this particular example two plot frames will be produced. See Section 8.6 for details on the $PLTDAT namelist.)

(The computer will now generate the following message)

1 GO/ERASE
2 GO/SAVE
3 EXIT
ENTER OPTION

(The user should respond with)

1 ↓

(The above message may be repeated several times, each time the user should type)

1 ↓

(until a picture similar to Figures 8.1 and 8.2 comes on the screen. At this point the user can study the picture and get a hard copy, if one wishes, by pressing the "copy" button. (Make sure the hard copy machine is turned on and warmed up. After leaving, turn the hard copy machine off. To see the next frame the user may type)

1 ↓

(The process is repeated until all frames have been seen. After this type)

3 ↓

COMMAND LOGOUT

(Push the screen reset button and the process is complete.)

8.8 Program SEEIT

8.8.1 Introduction

The main input to program MATCH3 is a disk file called TAPE10. TAPE10 is coded in binary format. If the user wants to examine TAPE10 before running MATCH, one could include the following two control cards:

```
COPYSBF,TAPE10,OUTPUT
REWIND,TAPE10.
```

These two control cards will transfer the data from TAPE10 onto the printer. However, the data printed would be in the binary format. Program SEEIT allows the user to obtain a printout of the data on TAPE10 in the normal floating point format.

8.8.2 Input to Program SEEIT

Usually the input to program SEEIT is a binary formatted tape (TAPE10) from program COMBS or program DECODG. However, input to SEEIT may be any file with the same format as TAPE10 created by COMBS or DECODG. This format being:

```
10A10,8I2    (First Record)
     6020    (All Following Records)
```

If program MATCH3 can use a file as input, then program SEEIT can "SEE' this file.

8.8.3 Output Of Program SEEIT

Program SEEIT (after the first heading record) reads two records at a time. Thus, it reads in twelve values at a time (each record format is 6020). With TAPE10 created by COMBS the last five of these values are zero. Program MATCH3 does not use these last five values. The printed output consists of the first seven values of each two record set. The format of the printed output is:

(E14.5,5X,E11.3,1X,E11.3,5X,E11.3,1X,E11.3,5X,E11.3,1X,E11.3)

8.8.4 Using Program SEEIT

Using program SEEIT is very simple. Three extra control cards (arrows show the extra cards that are needed) are needed to run SEEIT. An example follows.

```
JOB CARD
ATTACH,SEE,SEEIT,ID=ELTERMAN,MR=1. ←
.
.
.
.
.
.
.
.
.
.
COMBS.
SEE,TAPE10. ←
REWIND,TAPE10.←
MATCH3.
```

The "REWIND,TAPE10" card is not always needed, but cannot hurt.

9.0 GENERAL INSERTION LOSS PROGRAMS

9.1 Introduction

GILPM is an acronym for general insertion loss plots (modified version). The original version has since been phased out, but the letter M at the end remains.

Whereas, program COMBS is designed to do a mathematical analysis of two to six combs acoustically in series having one comb as an input port and one comb as an output port, program GILPM is capable of analyzing the insertion loss (as a function of frequency) of two to eight pairs of combs. The analysis is performed acoustically in parallel and electrically in series, parallel, or any combination thereof having one input port and one output port. The output connection is assumed to be identical to the input connection. The user has total generality in describing the matching network of the device.

Program GILPM accepts input from program COMBS and optionally from programs MATCH and RIND. Besides printed output, program GILPM produces TAPE24 which can be used (indirectly) by program GRAF or GRAFTEK to produce insertion loss plots.

Program RIND provides additional versatility to program GILPM by providing variable inputs.

Program GETMATD allows the input specifications to program GILPM to be checked for errors before actually trying to run GILPM which is a large and time-consuming program.

9.2 Input to Program GILPM

9.2.1 Input Impedance Tapes

For each comb on the device there must be a tape describing the input impedances (as a function of frequency) of the comb. The range of frequencies at which the input impedances are calculated is variable, however, it must be consistent for all combs in a particular device. The input impedance tapes are created by program COMBS. Usually each impedance tape is created with a separate COMBS run. Rarely is program COMBS executed more than once during a separate run. Program COMBS and program GILPM are also rarely run at the same time. Thus, to analyze a four comb device, for example, using program GILPM would require five separate runs (four COMBS runs and a GILPM run). As output, each run of COMBS produces TAPE10. Each version of TAPE10 (created by the different COMBS runs) must be saved as a permanent file. For example, in a four comb device the four catalog cards (in each separate run of COMBS) might look as follows.

```
CATALOG,TAPE10,TAPE10ONE,ID=ELTERMAN,MR=1.

CATALOG,TAPE10,TAPE10TWO,ID=ELTERMAN,MR=1.

CATALOG,TAPE10,TAPE10THREE,ID=ELTERMAN,MR=1

CATALOG,TAPE10,TAPE10FOUR,ID=ELTERMAN,MR=1.
```

The actual names of the tapes used by program GILPM, however, are TAPE11, TAPE12, TAPE13...TAPE18. Thus, for the above example, the following four cards would appear at the beginning of the GILPM run.

ATTACH,TAPE11,TAPE10ONE,ID=ELTERMAN,MR=1.

ATTACH TAPE12,TAPE10TWO,ID=ELTERMAN,MR=1.

ATTACH,TAPE13,TAPE10THREE,ID=ELTERMAN,MR=1.

ATTACH,TAPE14,TAPE10FOUR,ID=ELTERMAN,MR=1.

Program COMBS, as used here, analyzes two combs acoustically in series with no matching network. TAPE10 contains the input impedance values of the first comb at each frequency and the output impedance values of the second comb at each frequency. Program GILPM uses only the input impedances.

9.2.2 TAPE22

TAPE22 does not have to exist in order for program GILPM to run. Often program MATCH3 is run before program GILPM. If program MATCH3 has been run then TAPE22 will exist. Otherwise TAPE22 will not exist. Program GILPM produces TAPE24 as output. TAPE24 is the same format as TAPE22 and can be used by program GRAF or GRAFTEK (the name must be changed from TAPE24 to TAPE22 before program GRAF or program GRAFTEK can run it). If TAPE22 exists then program GILPM takes the plot identification off TAPE22 and copies it on TAPE24. If TAPE22 does not exist, then TAPE24 will have the following default plot identification.

ROGER COLVIN.

9.2.3 TAPE25

TAPE25 is created by program RIND. TAPE25 contains values of Q1, Q2, Q3, Q4, Q5, Q6, Q7, and Q8 at each frequency. TAPE25 does not have to exist as long as the input data to program GILPM does not reference any Q value. If TAPE25 does not exist and a Q value is referenced, then the program will abort. Q values are further described in Section 9.3.

9.2.4 The $GIL Namelist

9.2.4.1 The $GIL Namelist Variables and Their Types

NUMZINS:	integer
NUMMATS	integer

LO	real
RG	real
VG	real
Cn	real array of length 8
Rn	real array of length 8
Ln	real array of length 8.
PRNT	logical

9.2.4.2 The $GIL Namelist Variables and their Defaults

NUMZINS	2
NUMMATS	3
LO	0.
RG	50.
VG	1.
Cn	1.E-12, 1.E-12.....1.E-12
Rn	0.,0.,0.,0.,0.,0.,0.,0.
Ln	0.,0.,0.,0.,0.,0.,0.,0.
PRNT	.False.

9.2.4.3 The $GIL Namelist Variables and their Functions

NUMZINS: The number of combs to be analyzed. There are NUMZINS tapes created by program COMBS.

NUMMATS: The number of matrices to be created and have their determenents calculated. NUMMATS almost always equals NUMZINS + 1.

LO: Series tuning inductor (see Section 9.3).

RG: Generator characteristic impedance (see Section 9.3)

VG: Generator voltage (see Section 9.3).

L(n): Tuning inductor associated with the n'th comb (see Section 9.3).

C(n): Parasitic shunt capacitance associated with the n'th comb (see Section 9.3).

R(n): Parasitic series resistance associated with the n'th comb (see Section 9.3).

PRNT: If PRNT = .True., the values of each matrix at the first frequency are printed out.

9.3 Creating the Matrices

At each frequency there are NUMMATS complex matrices created and solved. Each matrix mathematically represents an electronic circuit consisting of transducers, capacitors, resistors and inductor. The size and general format of "NUMMATS" matrices are defined once by the user. The specific entry values are recalculated at each new frequency. The "NUMMATS" matrices may vary in size from a 2 x 2 matrix (four entries) to a 16 x 16 matrix (256 entries). The general description of each entry consists of from zero to sixteen descriptors. An entry described by zero descriptors represents a matrix value of zero. Each non-zero descriptor is described by exactly five characters. There are two possible formats for these five character descriptors. They are as follows:

$$\begin{bmatrix} + \\ - \end{bmatrix} \quad \begin{bmatrix} \mathrm{SN} \\ \mathrm{SM} \\ \mathrm{SI} \end{bmatrix} \quad \begin{bmatrix} \mathrm{VG} \\ \mathrm{LO} \\ \mathrm{RG} \end{bmatrix}$$

or

$$\begin{bmatrix} + \\ - \end{bmatrix} \quad \begin{bmatrix} \mathrm{SN} \\ \mathrm{SM} \\ \mathrm{SI} \end{bmatrix} \quad \begin{bmatrix} \mathrm{C} \\ \mathrm{R} \\ \mathrm{L} \\ \mathrm{Q} \\ \mathrm{A} \\ \mathrm{B} \end{bmatrix} \quad \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{bmatrix}$$

The following formats, however, are illegal even though they are syntactically described by the second format above.

$$\begin{bmatrix} + \\ - \end{bmatrix} \quad \begin{bmatrix} \mathrm{SI} \end{bmatrix} \quad \begin{bmatrix} \mathrm{Q} \end{bmatrix} \quad \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{bmatrix}$$

and

$$\begin{bmatrix} + \\ - \end{bmatrix} \quad \begin{bmatrix} \mathrm{SI} \\ \mathrm{SM} \end{bmatrix} \quad \begin{bmatrix} \mathrm{A} \\ \mathrm{B} \end{bmatrix} \quad \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{bmatrix}$$

To form a valid descriptor using the first format (containing three brackets) take any symbol in the first bracket, followed by any two letter set in the second bracket, followed by any two letter set in the third bracket.

To form a valid descriptor using the second format (containing four brackets) take any symbol in the first bracket, followed by any two letter set in the second bracket, followed by any letter in the third bracket, followed by any number in the fourth bracket.

To find out which of these descriptors are not legal check to see if they are syntactically described by the two "illegal" formats. In other words, a descriptor is legal if it is described by either the first or second formats and not described by either the third or fourth formats.

Thus, there are a total of

$(2x3x3) + (2x3x6x8) - (2x1x1x8) -(2x2x2x8) = 18 + 288 - 16 - 64$

$= 226$ legal descriptors.

Each of the following are examples of legal descriptors:

+SNVG

-SML3

+SIQ2

+SNRG

Each of the following are examples of illegal descriptors:

+SNV3

-SMC9

+SQA3

$SMVG

+SIQ4

-SIA3

+SMB4

If the program finds even one illegal descriptor, execution is aborted. (For this reason the input should first be checked by program GETMATD, which will be described in following sections.)

9.3.1 Definitions of Descriptor Components

The following components are defined by the $GIL Namelist and are constants (i.e. not frequency dependent).

VG: Generator voltage (usually normalized to 1).

RG: Generator characteristic impedance (usually 50 ohms).

LO: Series tuning inductor.

Cn: Parasitic shunt capacitance associated with the n'th comb.

Rn: Parasitic series resistance associated with the n'th comb.

Ln: Tuning inductor associated with the n'th comb, where n = 1, 2, 3, ... 8.

The following components are defined by the input tapes from program COMBS. They are frequency dependent.

An: The real part of the input impedance associated with the n'th comb.

Bn: The imaginary part of the input impedance associated with the n'th comb, where n = 1, 2, 3, ... 8.

(Engineer's Note: ZTn = An + iBn)

The following component is defined by the input tape (TAPE25) from program RIND. It may or may not be dependent on frequency depending on the version of RIND used (program RIND is described in following sections).

Qn: In general Qn is an arbitrary number representing any electrical device the user may wish to add. The values of Qn come from TAPE25 which is created by program RIND. Program RIND was not designed to be a fixed program. The user may alter program RIND to simulate any electrical component he desires that cannot be simulated from existing components.

The following three components determine whether the descriptor represents a real or imaginary number and whether or not the descriptors are inverted.

SN: SN stands for "Not Multiplied by S". The value of SN is always equal to one (SN=1).

SM: SM means that the following component of the descriptor is multiplied by S. S is a frequency dependent imaginary number. The value of S is $i2\pi f$ ($S=i2\pi f$).

SI: SI means that the following component of the descriptor is inverted and then multiplied by 1/S.

Examples:

Descriptor Notation	Mathematical Notation
+SNR3	R_3
+SMR3	$i2\pi fR_3$
+SIR3	$\frac{1}{i2\pi fR_3} = i\frac{-1}{2\pi fR_3}$

9.4 A Real Life Example

The following example was actually programmed. This example cannot be understood unless the reader is familiar with electrical engineering.

Figure 9.1 shows two transducers in parallel, together with matching inductors and parasitic elements. The output is assumed identical to the input and is not shown.

Figure 9.2 shows the same electrical device redrawn in a form which is more convenient for circuit analysis.

The following is the mathematical analysis of these circuits.

$$V_g - R_g I_1 - SL_o I_1 - SL_1 I_1 - \left(\frac{1}{SC_1}\right) I_1 + \left(\frac{1}{SC_1}\right) I_2 - R_g I_3 - SL_o I_3 + 0I_4 = 0$$

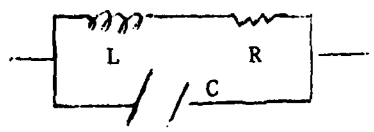
$$0 + \left(\frac{1}{SC_1}\right) I_1 - \left(\frac{1}{SC_1}\right) I_2 - R_1 I_2 - Z_{T1} I_2 + 0I_3 + 0I_4 = 0$$

$$V_g - R_g I_1 - SL_o I_1 + 0I_2 - R_g I_3 - SL_o I_3 - SL_2 I_3 - \left(\frac{1}{SC_2}\right) I_3 + \left(\frac{1}{SC_2}\right) I_4 = 0$$

$$0 + 0I_1 + 0I_2 + \left(\frac{1}{SC_2}\right) I_3 - \left(\frac{1}{SC_2}\right) I_4 - R_2 I_4 - Z_{T2} I_4 = 0$$

$$\begin{bmatrix} \left(R_g + SL_o + S_{L1} + \frac{1}{SC_1}\right) & \left(\frac{-1}{SC_1}\right) & \left(R_g + SL_o\right) & 0 \\ \left(-\frac{1}{SC_1}\right) & \left(\frac{1}{SC_1} + R_1 + Z_{T1}\right) & 0 & 0 \\ \left(R_g + SL_o\right) & 0 & \left(R_g + SL_o + SL_2 + \frac{1}{SC_2}\right) & \left(\frac{1}{-SC_2}\right) \\ 0 & 0 & \left(-\frac{1}{SC_2}\right) & \left(\frac{1}{SC_2} + R_2 + Z_{T2}\right) \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \end{bmatrix} = \begin{bmatrix} V_g \\ 0 \\ V_g \\ 0 \end{bmatrix}$$

The above equations have assumed ideal inductors. However, it was desired to use the real model for inductors L_1 and L_2. A possible circuit model for a real inductor (to replace the ideal inductor) is



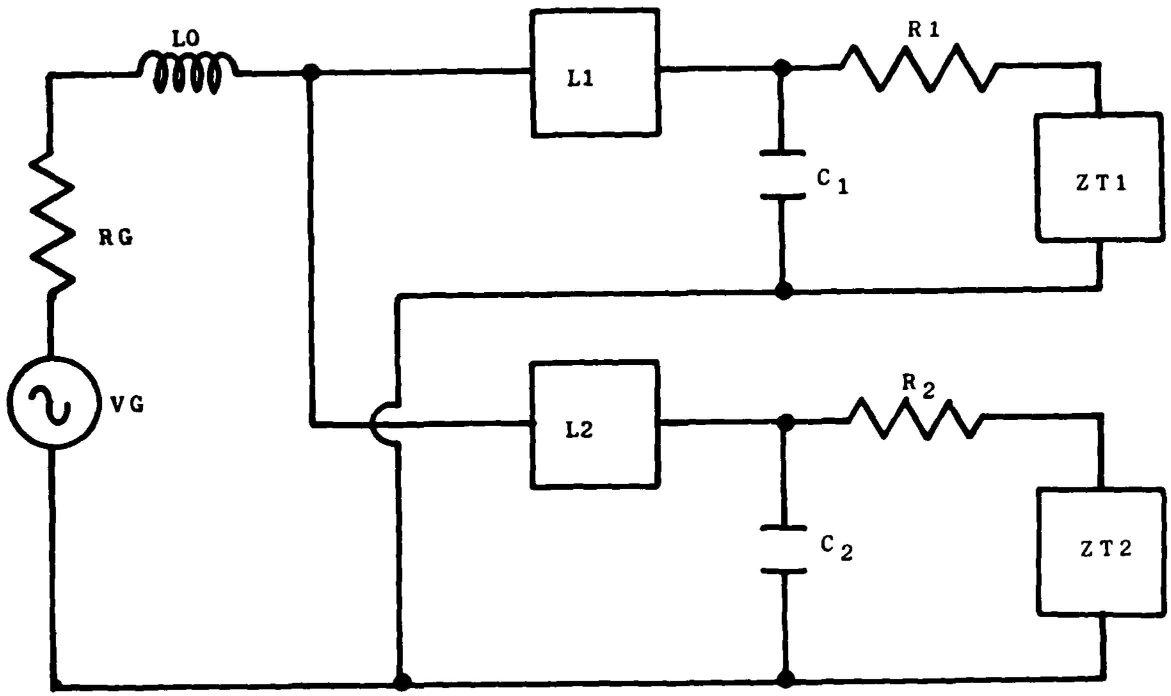


Figure 9.1 Two Transducers in Parallel With Matching Inductors and Parasitic Elements

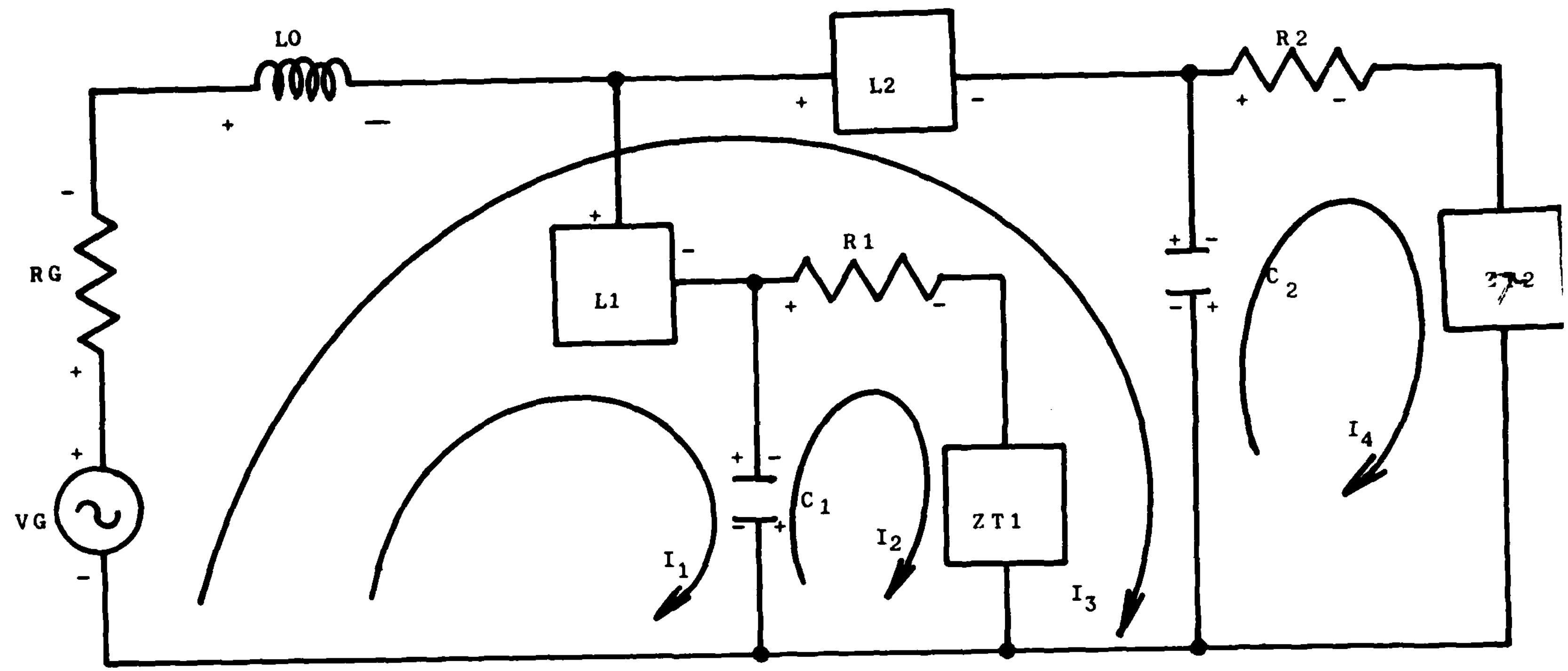


Figure 9.2 The Same Electrical Device as shown in Figure 9.1
Redrawn for Convenient Circuit Analysis

where $C = \dfrac{1}{L\left[(2\pi\ SRF)^2 + (^R/_L)^2\right]}$

$$R = \frac{2\pi f L}{Q}$$

Q = finite Q of the real inductor

and SRF = self resonant frequency of the real inductor.

The impedance (Z_L) of a real inductor corresponding to the above model is given by

$$Z_L = \frac{1}{C}\ \frac{iw + R/L}{iwR/L + \left(^1/_{LC} - w^2\right)}$$

$$Z_L = \frac{1}{C}\ \frac{R/(L^2C)}{w^2\left(\frac{R}{L}\right)^2 + \left(\frac{1}{LC} - w^2\right)^2} + \frac{iw}{c}\ \frac{\frac{1}{LC} - w^2 - \left(\frac{R}{L}\right)^2}{w^2\left(\frac{R}{L}\right)^2 + \left(\frac{1}{LC} - w^2\right)^2}$$

$Z_L = Q_1 + iwQ_2$, where $S = iw = i2\pi f$

In the matrix equations below SL_1 is replaced by $Q_1 + SQ_2$. (Q_1 and Q_2 are computed and passed from RIND to GILPM.)* Q_1 is the real part of the impedance of the real inductor and Q_2 is the imaginary part of the impedance of the real inductor. (Note that Q_2 as passed from RIND has not yet been multiplied by S.) It then follows that SL_2 is replaced by $Q_3 + SQ_4$. (Q_3 and Q_4 are also passed from RIND).

(Note that RIND is not a set unchangeable program. Program RIND may be changed by the user so that the values of Q_1, Q_2, $Q_3 \ldots Q_8$ represent any mathematical equations desired by the user. In this particular example program RIND had to be specifically programmed so that the values $Q_1 + SQ_2$ and $Q_3 + SQ_4$ represented real inductors. The values Q_1, Q_2, Q_3 and Q_4 are frequency dependent.)

Also in the matrix equations below Z_{T1} is replaced by $A_1 + iB_1$ and Z_{T2} is replaced by $A_2 + i B_2$.

*These Q_n are different from the finite Q values of the real inductors found in namelist $QIND.

Z_{Tn} is a complex number representing the impedance of the n'th transducer. When Z_{Tn} is replaced by $A_n + iB_n$ the A_n represents the real part of the impedance and B_n represents the imaginary part of the impedance (see Section 9.3.1).

$$\begin{bmatrix} \left(R_g + SL_o + Q_1 + SQ_2 + \frac{1}{SC_1}\right) & \left(\frac{1}{-SC_1}\right) & \left(R_g + SL_o\right) & 0 \\ \left(-\frac{1}{SC_1}\right) & \left(\frac{1}{SC_1} + R_1 + A_1 + B_1\right) & 0 & 0 \\ \left(R_g + SL_o\right) & 0 & \left(R_g + SL_o + Q_3 + SQ_4 + \frac{1}{SC_2}\right) & \left(\frac{-1}{SC_2}\right) \\ 0 & 0 & \left(\frac{-1}{SC_2}\right) & \left(\frac{1}{SC_2} + R_2 + A_2 + B_2\right) \end{bmatrix} \begin{bmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \end{bmatrix} = \begin{bmatrix} V_g \\ 0 \\ V_g \\ 0 \end{bmatrix}$$

This matrix equation is of the form $[A]\,\vec{X} = \vec{B}$, where

$$[A] = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

Let $D = \mathrm{DET}[A]$

$$N_1 = \mathrm{DET} \begin{bmatrix} a_{11} & V_g & a_{13} & a_{14} \\ a_{21} & 0 & a_{23} & a_{24} \\ a_{31} & V_g & a_{33} & a_{34} \\ a_{41} & 0 & a_{43} & a_{44} \end{bmatrix} \quad \text{and}$$

$$N_2 = DET \begin{bmatrix} a_{11} & a_{12} & a_{13} & V_g \\ a_{21} & a_{22} & a_{23} & 0 \\ a_{31} & a_{32} & a_{33} & V_g \\ a_{41} & a_{42} & a_{43} & 0 \end{bmatrix}$$

Then $\left|I_2\right|^2 = \frac{\left|N_1\right|^2}{\left|D\right|^2}$ = ICAP (1), and

$$\left|I_4\right|^2 = \frac{\left|N_2\right|^2}{\left|D\right|^2} = \text{ICAP (2).}$$

(Note: N_1, N_2, and D are complex numbers; ICAP (N) is defined below.)

$$\text{Transducer efficiency} = \frac{2RG}{VG}\left[\left|I_2\right|^2 ReZ_{T1} + \left|I_4\right|^2 ReZ_{T2}\right]$$

$$\text{Insertion Loss (in dB)} = -10 \text{ Log}_{10} \text{ (Transducer efficiency)}^2$$

The actual routine that computes the insertion loss (subroutine ILCALC) is not a fixed part of program GILPM, but rather a changeable subroutine that is usually part of the run deck (see Appendix A).

Input to subroutine ILCALC is arrays ICAP(8) and REZT(8), where

ICAP(N) = $\frac{\text{The magnitude of the determinant of the n'th matrix}}{\text{The magnitude of the determinant of the NUMMAT'th matrix}}$

(NUMMATS is defined in Section 9.2.4.3)

REZT(N) = A_N

(A_N is defined in Section 9.3.1)

ICAP (N) and REZT (N) are automatically passed from GILPM.

Output of program ILCALC, returned to GILPM, is the variable "IL" containing the insertion loss.

The cards of subroutine ILCALC may be changed by the user to correspond with the particular device one is analyzing.

9.5 The Run Deck of the Real Life Example

See Appendix A for the data deck used to program the example described in this section.

Also shown in the Appendix is a GETMATD run. Program GETMATD allows the user to check the syntax of one's program without having to run program GILP (a large and time-consuming program).

10.0 GENERAL TEKTRONIX INTERFACE PROGRAMS

10.1 Introduction

The advantages of using the Tektronix machine for plotted output (instead of or in addition to the Cal-Comp plotter) is that turn-around time is greatly improved. (For a general discussion of the Tektronix machine see Section 8.7.1.)

Suppose a user has a fully working program that produces a plot on the Cal-Comp plotter. Everything works fine, except that the user has to wait an average of two days for his plots. One finds the slow turn-around time for his plots is slowing down progress on all related work, and decides to use the Tektronix machine.

Even in the most simple case, interfacing a program to the Tektronix machine will require several small modifications such as the addition of calls to subroutine SHOWNGO and connecting and disconnecting of output. The user must also make sure that all input is available from the Tektronix machine. This means that all input, including card input, must be saved on the disk as a permanent file.

Often, however, it is just about impossible to modify the program to work on the Tektronix machine. The reason for this is that a maximum memory of 65000 (octal) words are allotted to the Tektronix machine. After subtracting from this the size of the Fortran library and the size of the Tektronix library, the user will find that there is a very limited program memory. It is difficult to modify a program 200000 (octal) words long, for example, to fit in a memory space of approximately 30000 (octal) words. If it can be done at all, it will take a complicated system of overlays and/or disk files to do it.

The programs discussed in this section are designed to provide a simple way for the user of any plotting program, of any size, to receive plots on the Tektronix machine. The job may be run on BATCH using a deck only slightly different from the original deck. The user may receive plots from both the Tektronix machine and the CAL-Comp plotter by submitting a single BATCH job.

10.2 Simplified Discussion Of How The Tektronix Interface System Works

The principle behind the Tektronix Interface System is not to modify the original program to run on the Tektronix machine, but rather to modify the original program so that it produces a file which contains a program that can be run on the Tektronix machine. Consider the following two programs:

```
PROGRAM LARGE (Input,Output)
DIMENSION K(100000)
DO  10  J=1, 100000
K(J) = J
CONTINUE
A=3.
B=6.
CALL PLOTID3(30H LARRY ELTERMAN TELEPHONE X4904, 500., 11., 1.)
CALL PLOT (A,A,3)
CALL PLOT (B,A,2)
CALL PLOT (B,B,2)
CALL PLOT (A,B,2)
CALL PLOT (A,A,2)
CALL ENDPLT
END

PROGRAM SMALL (INPUT,OUTPUT)
CALL PLTID3 (30H LARRY ELTERMAN TELEPHONE X4904, 500., 11., 1.)
CALL PLOT (3.,3.,3)
CALL PLOT (6.,3.,2)
CALL PLOT (6.,6.,2)
CALL PLOT (3.,6.,2)
CALL PLOT (3.,3.,2)
CALL ENDPLT
END
```

Program LARGE is very large. It is over 100000 (decimal) words long. It is much too large to run on the Tektronix machine. Program SMALL, on the other hand, is very small. It is easily small enough to run on the TEKTRONIX machine. Yet, as far as plotting is concerned, programs SMALL and LARGE produce the exact same results (both draw a three inch square).

These two programs illustrate the basic principle behind the Tektronix Interface System. The plotting instructions (only) are extracted from the original program during execution and put on a separate file to be compiled and run on the Tektronix machine. These extracted plotting instructions usually take up less core than the original program and can usually be run on the Tektronix without exceeding memory.

10.3 Detailed Discussion Of How The Tektronix Interface System Works

There are two pieces of software in the Tektronix Interface System. The first is program TEK and the second is a system of subroutines referred to as the TEK Interface Library.

The purpose of program TEK is to take an original program designed to produce plots on the Cal-Comp plotter and change the names of all the plotting routines. It also adds TAPE94, 95, and 96.

Example: Two versions of program EXAMPLE are shown. The first version is the original version. The second version shows program EXAMPLE after it has been processed by program TEK.

```
Program EXAMPLE(INPUT,OUTPUT)
CALL PLTID3(30H LARRY ELTERMAN TELEPHONE X4904, 500., 11., 1.)
CALL PLOT(2.,2.,3)
CALL PLOT(2.,4.,2)
CALL SYMBOL(10.5,1.13,.14,10.,90.,41)
CALL ENDPLT
END
```

```
PROGRAM EXAMPLE(INPUT,OUTPUT,TAPE94,TAPE95,TAPE96)
CALL PLT1D9(30H LARRY ELTERMAN TELEPHONE X4904, 500., 11., 1.)
CALL PLO9(2.,2.,3)
CALL PLO9(2.,4.,2)
CALL SYMBO9(10.5,1.13,.14,10.,90.,41)
CALL ENDPL9
END
```

The following is the list of plot routines that have their name changed and what they are changed to:

1. DQPLOT ⟶ DQPLO9
2. PLT1D3 ⟶ PLTID9
3. AXIS ⟶ AXI9
4. LINE ⟶ LIN9
5. PLOT ⟶ PLO9
6. SYMBOL ⟶ SYMBO9
7. NUMBER ⟶ NUMBE9
8. ENDPLT ⟶ ENDPL9

When running the version of the program modified by program TEK, both the ONLINEPEN (or OFFLINEPEN) library and the TEK Interface Library must be loaded into the system.

When a program calls one of the above routines (with a nine at the end) it is (from the main program's point of view) just as if it called the original plotting routine.

Let's consider an example. Suppose program NUMPLOT originally calls the plotting routine number. After being processed by program TEK it will call routine NUMBE9 instead of NUMBER. When executing this program the TEK Interface Library will be in the system. What happens when this routine is called? Let's examine program NUMBE9 (the simplest of the TEK Interface Library routines).

```
SUBROUTINE NUMBE9        74/74    OPT=1                    FTN 4.6+428

 1               SUBROUTINE NUMBE9(X,Y,HT,FPN,TH,N)
 2               DIMENSION M(30)
 3               CALL NUMBER(X,Y,HT,FPN,TH,N)
 4               ENCODE(40,1,M)X
 5         1     FORMAT(6X,*CALL NUMBER(*,028,*B,*)
 6               ENCODE(110,2,M(5))Y,HT,FPN,TH,N
 7         2     FORMAT(020,*B,*,020,*B,*,020,*B,*,020,*B*,020,*B)*)
 8               CALL TEXT47G(150,M)
 9               RETURN
10               END
```

Note the following:

Line 1: The input parameter list is the same for routine NUMBE9 as it is for routine NUMBER.

Line 3: The plotting routine NUMBER is called with exactly the same values that the original program would have directly called it with had it not been processed by program TEK. From the point of view of the plotter it is receiving the same instructions it would have from the original program. Note: The ONLINEPEN (or OFFLINEPEN) library must also be loaded into the system.

Lines 4,5,6,7: The call to NUMBER is converted to alphanumeric code and stored in the array M. All variables are converted to octal constants to assure no loss in accuracy.

Line 8: The array M is written out to disk file TAPE94 in a format that can be interpreted by the Fortran compiler.

Line 9: Execution is returned to the main program. None of the input parameter values have been altered (the same as if only routine NUMBER had been called). Thus, from the main program's point of view (i.e. input, output) calling routine NUMBE9 is no different than calling routine NUMBER.

The other TEK Interface Library routines are similar to NUMBE9. Every time a plotting routine is executed the same plot call is placed on TAPE94. After the main program has finished execution TAPE94 contains (in alphanumeric source code) all the calls to plot that occurred during execution. The user can then run this created program on the Tektronix terminal.

The following is a brief discussion of the other routines in the TEK

Interface Library:

PLTID9: This routine must be the first routine called in the main program. Besides writing out the call to PLTID3 on TAPE94 it also sets up text on TAPE94 so that when the program on this tape is executed the Tektronix machine asks the user what factor one wants to use. (A factor value of .5, for example, makes the plotted picture half the size it usually would be with Factor = 1.) This is useful when the size of the plot is too big to put on the screen with the normal factor value of one. PLTID9 also sets up the dimension statement "DIMENSION X(1003),Y(1003)" which is necessary to routine LIN9; sets up the format statement "1 FORMAT(20I20)" which is needed by routine LIN9; and the format statements "2 FORMAT(F10.5)", and "3 FORMAT(1X, 13H WHAT FACTOR?)*) which are used in determining the value FACTOR at the beginning of TAPE94. Finally, since it is theoretically legal (though a seldom used option) to call PLTID3 more than once during execution of a program, routine PLTID9 uses TAPE95 to check if it has been already called. The two cases must be handled differently. TAPE95 also keeps track of SHOWNGO coordinates (variables A and B) and the line number to use when a continue statement is needed (variable KONTINU).

AXI9: Similar in implementation to routine NUMBE9. AXI9 is somewhat more complex than NUMBE9 due to the calling of routine MES457Q which is used to handle the variable length message which may be plotted alongside the axis.

PLO9: There are basically two kinds of calls to routine PLOT. The first kind repositions the pen. It may or may not write a line as it repositions depending on what the user specifies. The second kind of call to plot repositions the origin. When this kind of command is called, the new origin is saved on TAPE95. Otherwise PLO9 is similar in implementation to NUMBE9.

SYMBO9: Very similar in implementation to AXI9.

LIN9: A call to plotting routine LINE can make reference to two arrays which may contain a thousand (or even more) points each. This presents a problem. How do you store these points on the source file, TAPE94?

There are two possible solutions. The first is to break up the call to LINE into a series of calls to PLOT, one call to PLOT for each point in the referenced array. The problem with this method is that after a few calls to LINE the Binary Run File compiled from TAPE94 may be too big to run on the Tektronix machine.

With the second method (the one actually used) LIN9 takes up to 1003 points stored in each of the referenced arrays and stores them on TAPE96.

This storing on tape takes place in lines 12, 13, 14 and 15 of routine LIN9. LIN9 then writes the instructions on TAPE94 for obtaining these points from TAPE96 and executing routine LINE. The instructions written on TAPE94 are of the following format:

```
      DO  10  J = 1, 150
      READ(96,1)X(J),Y(J)
10    CONTINUE
      CALL LINE (appropriate parameter list)
```

DQPLO9: This routine is broken up into a series of calls to PLTID9, SCALE, AXI9, and LIN9.

ENPL9: This must be the last plotting routine called by the main program. First a call to ENDPLT is written on TAPE94, then ENDPLT is called ending execution.

TEXT47G: This routine takes the text coded in array M (N characters long) and writes it onto TAPE94. It breaks the text into records of length 72 or less and places the number 1 in the sixth column to specify a continuation card when more than one record is needed to code a Fortran statement.

MES457Q: This routine was written to solve a tricky little problem. Consider routine AXI9. The message the user wishes plotted out along the axis is passed to routine AXI9 through the array variable MES. MES is dimensioned to ten. However, because MES is not originally defined in subroutine AXI9, the dimensioned length of ten is only a dummy value. The actual length of the array is determined by the length of the message and this value is not passed to AXI9. The problem arises when trying to encode the message into array M. The exact length of the array must be known.

Routine MES457Q transfers the information in array MES (unknown length) to array L (known length). It either truncates the message or adds blanks, as necessary to make it exactly fill up array L.

Note that the plotting routines WHERE and SCALE are not changed (to SCAL9 and WHER9) since these routines produce no plotted output.

10.4 Using the Tektronix Interface System

10.4.1 Converting Original Program To Modified Form

The first step the user must accomplish is convert the original program to the modified version discussed in Section 10.3. To do this the user must have the source code (that is, Fortran statements) of the original program. This source code may either be in the form of physical cards or as a file on disk. Let's consider the case where the user has a program on cards. To create the modified version, the user might submit the following deck:

```
JOB CARD  CM=65000,  TIME=100.
REQUEST,LGO,*PF.
ATTACH,TEK,SAVETEK,ID=ELTERMAN,MR=1.
TEK.
FTN,SL,I=TAPE7.
REWIND,LGO.
CATALOG,LGO,MODIFIEDVERSION,ID=ELTERMAN,MR=1.
789
```

SOURCE PROGRAM DECK

```
6789
```

The output of program TEK is TAPE7 which contains the source code of the modified program. In the above example TAPE7 is compiled and the binary version of the modified program is saved on permanent file.

In the next example the user has the source file on disk as a permanent file.

```
JOB CARD  CM=65000,  TIME=100.
REQUEST,LGO,*PF.
ATTACH,TEK,SAVETEK,ID=ELTERMAN,MR=1.
ATTACH,SOURCE,SAVESOURCE,ID=ELTERMAN,MR=1.
TEK,SOURCE.
FTN,SL,I=TAPE7.
REWIND,LGO.
CATALOG,LGO,MODIFIEDVERSION,ID=ELTERMAN,MR=1.
6789
```

In the last example the source program is on cards and the source code of the modified program is also put on cards.

```
JOB CARD  CM=65000,  TIME=100.
ATTACH,TFK,SAVETEK,ID=ELTERMAN,MR=1.
TEK.
COPYBF,TAPE7,PUNCH.
789
```

SOURCE PROGRAM DECK

6789

In all the above examples "SAVETEK" is the binary version of program TEK.

10.4.2 Converting the Batch Run Deck

10.4.2.1 Two Example Run Decks

In the following examples assume that both the original and modified versions of a program are stored as binary (already compiled) files on disk. The TEK Interface Library must also be stored as a binary file.

The first example shows a possible deck setup used to run the original program (which produces an on line plot only).

```
JOB CARD  CM-120000, T=30.
ATTACH,ORIG,ORIGINALVERSION,ID=ELTERMAN,MR=1.
ATTACH,PEN,ONLINEPEN
LIBRARY,PEN
REQUEST,PLOT,*Q.
DISPOSE,PLOT,*OL.
ORIG.
789
```

INPUT DATA

6789

The next example shows how the run deck is modified so that the user can receive his plots on the Tektronix machine as well as from the Cal-Comp plotter

```
JOB CARD  CM=140000,  TIME=50.
ATTACH,MOD,MODIFIEDVERSION,ID=ELTERMAN,MR=1.
ATTACH,TEKLIB,TEKINTERFACELIB,ID=ELTERMAN,MR=1.
REQUEST,TAPE94,*PF.
```

```
REQUEST,LGO,*PF.
ATTACH,PEN,ONLINEPEN.
LIBRARY,PEN.
REQUEST,PLOT,*Q.
DISPOSE,PLOT,*OL.
LOAD(MOD)
LOAD(TEKLIB)
LDSET(Preset=Zero)
EXECUTE.
FTN,I=TAPE94.
CATALOG,LGO,TAPE94BINARY,ID=ELTERMAN,MR=1.
CATALOG,TAPE96,SAVETAPE96,ID=ELTERMAN,MR=1.
EXIT(U)
789
```

INPUT DATA

```
6789
```

The above deck setup will do everything the first deck setup will do, and in addition it will save the files TAPE94BINARY and SAVETAPE96, which can be used to produce a plot at the Tektronix machine. (The above deck setup will produce a Cal-Comp Plot identical to the one produced by the first run deck.)

10.4.2.2 Differences Between The Two Example Run Decks

A) The second deck uses more core memory. The reason for this is that it must accommodate the additions of TAPE94, TAPE95, and TAPE96 to program MOD as well as the loading of the Tek Interface Library. The exact amount of core that must be added has not yet been determined (an easy fact to figure out). However, 20000 (octal) extra words should cover all the overhead.

B) The second deck requires more time to run than the first deck. This is because routine PLO9, for example, takes more time to run than routine PLOT. The amount of extra time required depends on the amount of times plotting routines are called.

C) The modified version of the program is attached rather than the original version.

D) The TEK Interface Library is attached.

E) TAPE94 and LGO are requested as permanent files.

F) To run the program takes the four card sequence:

```
LOAD(MOD)
LOAD(TEKLIB)
LDSET(Preset=Zero)
EXECUTE.
```

G) After execution TAPE94 is compiled and saved as a permanent file.

H) TAPE96 is saved as a permanent file.

I) The card "EXIT(U)" follows the catalog TAPE96 card. If there is no call to routine LIN9 during execution of the modified program then TAPE96 will never be created. Trying to catalog a non-existent file causes an error condition. The EXIT(U) card keeps the run from terminating on the error condition.

10.4.3 Using The Tektronix Machine To Produce A Plot

Once the user has taken all preliminary steps as described in Sections 10.4.1 and 10.4.2, one is ready to receive the plots from the Tektronix machine. The user should then complete the following dialog with the Tektronix machine. In the following example underlined text represents text written on the Tektronix machine by the computer. Non-underlined text represents commands typed into the Tektronix machine by the user. Text in parenthesis represents the author's comments. ↓ means the user should press the carriage return on the Tektronix keyboard.

LOGIN,NAME,CODE,8614444,SUP
(The user must have one's own LOGIN name and code)

<u>COMMAND</u> SCREEN,80,66. ↓

<u>COMMAND</u> ATTACH,TEK,TEKLIB. ↓

<u>COMMAND</u> LIBRARY,TEK. ↓

<u>COMMAND</u> ATTACH,LGO,TAPE94BINARY,ID=ELTERMAN,MR=1. ↓

<u>COMMAND</u> ATTACH,TAPE96,SAVETAPE96,ID=ELTERMAN,MR=1. ↓

(SAVETAPE96 may not be cataloged (see Section 10.4.2). If this is the case then the above command may be omitted. If you give the above command and get an error message do not worry about this, go on to the next step.)

<u>COMMAND</u> LGO. ↓

(Wait for the computer to printout the following message.)

<u>WHAT FACTOR</u>?

.75 ↓

(The user may type in any appropriate factor, see Section 10.3 under the discussion of routine PLTID9.)

AD-A079 623 ANALYSIS AND COMPUTER SYSTEMS INC BURLINGTON MA F/G 17/1
DEVELOPMENT OF NUMERICAL TECHNIQUES AND COMPUTER SYSTEMS FOR DE--ETC(U)
AUG 79 L J ELTERMAN F19628-76-C-0257

UNCLASSIFIED RADC-TR-79-181 NL

3 OF 4
ADA 079623

(The computer will now type out the following message)

1 GO/ERASE
2 GO/SAVE
3 EXIT
ENTER OPTION

(A plot frame from the user's program may or may not appear with the above message. Every time the user wishes to see the next frame one should type)

1↓

(until he has seen all the frames one wishes to see, or until there are no more frames. At any time the user may obtain a hard copy of whatever is on the screen by pushing the copy button. If the user wishes to end execution one should type)

3 ↓

COMMAND LOGOUT

(Push the screen reset button and the job is terminated.)

10.5 Restrictions For Using The TEK Interface System

10.5.1 Restrictions On The User's Original Program

A) None of the program namelists, variables, commons, subroutines or functions may be named any of the following:

1. DQPLO9
2. PLTID9
3. AXI9
4. LIN9
5. PLO9
6. SYMBO9
7. NUMBE9
8. ENDPL9
9. TEXT47G
10. MES457Q

If the program does use any of the above names, they must be changed before the program can be modified. The above names have been chosen to reduce the likelihood of this happening.

B) No call to routine LINE may reference more than 1003 points (i.e. not more than 1003 points can be used to draw the line).

C) No single Fortran statement can be longer than eight lines long (seven continue cards).

D) The program cannot use TAPE94, TAPE95 or TAPE96 as input or output files.

E) PLTID3 or DQPLOT must be the first plotting routine called.

F) None of the plotted messages in the routines AXIS, SYMBOL, or DQPLOT may be more than 100 characters long.

G) ENDPLT must be the last plotting routine called.

H) The first card of the program must be a program card.

Example: Program MACK(Input, Output)

The first card must not be a comment card nor anything else.

10.5.2 Other Restrictions

The program compiled from TAPE94, the Tektronix Library and the TEK Interface Library together must be less than 65000 (octal) words. In all but the most complex plots this will be the case.

11.0 SUPPLEMENTAL SYSTEM PROGRAMS

11.1 Program RATIO

11.1.1 Introduction

Program RATIO is a program to find the ratio of metal to the total length of a two comb transducer. The ratio is calculated along a horizontal cross section of the transducer (see Figure 11.1). Program RATIO receives its input directly from program CONVERT.

11.1.2 Input To Program Ratio

Input to program RATIO consists of TAPE9, TAPE28, TAPE29, the RATIO title card and the $RATIO namelist.

TAPE9, TAPE28 and TAPE29 are created by program CONVERT. If CONVERT has been run properly prior to the running of RATIO, then the above tapes (actually disk files) will be available to program RATIO.

The value NCOMBS (received from the $CONVERT namelist via TAPE28) must be equal to two or the program will abort.

The RATIO title card goes before the $RATIO namelist card.

The $RATIO namelist contains the single variable ZZO (default = 9.E-3) which specifies the horizontal distance between the center of the zero'th gap of the first comb to the center of the zero'th gap of the second comb.

11.1.3 Information Provided By Program RATIO

Program RATIO provides the following information to the user.

A) The number of fingers in the first comb.
B) Amount of space in the first comb.
C) Amount of metal in the first comb.
D) Length of the first comb.
E) Amount of space in the second comb.
F) Amount of metal in the second comb.
G) Length of the second comb.
H) Total space in both combs.
I) Total metal in both combs.
J) Total length of both combs.
K) Distance between the two combs.
L) Ratio in first comb metal/length.
M) Ratio in second comb metal/length.
N) Ratio (total metal)/(total length of both transducers).
O) The number of fingers in the second comb.
P) Distance from the outer edge of the first comb to the 0'th gap of the first comb.
Q) Distance from the outer edge of the first comb to the 0'th gap of the second comb.

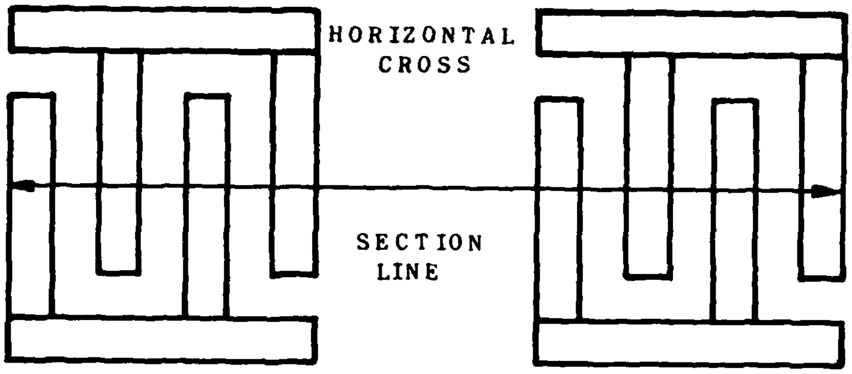


Figure 11.1 Ratio Example

In the above example the ratio of metal to total length along a horizontal cross section is 8/19 = .42105.

11.1.4 Running Program RATIO

(See example in Appendix)

11.2 Program MERGEG

Program MERGEG was slightly modified by ACSI. The logical variable FLIP was added to the $MERGE namelist. If FLIP = .False., the program behaves as always. (FLIP = .False. is default). If FLIP = .True., then the input data forming the complex admittance matrices is flipped.

Example: Let the following be one set of input to program MERGEG.

FREQ. Complex1 Complex2 Complex3

If FLIP = .False. the matrix will be formed as follows:

Complex1	Complex2
Complex2	Complex3

If FLIP = .True. the matrix willl be formed as follows:

Complex3	Complex2
Complex2	Complex1

11.3 Program LETTER

11.3.1 Introduction

Program LETTER uses information from program HUGHES to produce a form letter. The letter gives such information as the number of tapes to be sent to the Hughes Company, the names of the tapes and the contract they are sent under.

11.3.2 Running Program LETTER

(See Appendix for sample run).

11.3.3 Example Letter

The following is an example of a letter produced by program LETTER:

Department of the Air Force
HQ. Rome Air Development Center (AFSC)
Deputy for Electronic Technology
Hanscom AFB, Massachusetts 01731

Reply to
Attn Of : EEA

Subject : Shipment of GFP Tapes

To : PPUA (O. PAPE/STOP 25)

1. Tapes Generated: 32ILL2 With 3 Transducers on This Tape.

2. The above mentioned tape is to be shipped on or about this date to Hughes Aircraft Company under Contract CC-XX384. Each tape produces 1 master.

Andrew J. Slobodnik, Jr.
Antenna and RF Components Branch
Electromagnetic Sciences Division

Cy to: SUO(A. Callahan)

11.4 Program UNDO

11.4.1 Introduction

The purpose of program UNDO is to be able to take an Ho[n] deck (see Section 3.4) that has been diffraction corrected and to "undo" the diffraction correction. Program UNDO can accept, as input, any set of cards that program CONVERT can accept (see Section 3.2). However, within that set of cards program UNDO only removes diffraction correction on Ho [N] or Ho[NTAP] decks. (An Ho[N] deck that has been diffraction corrected is sometimes called an HoD[N] deck. Diffraction corrected decks are created by program SAL.)

11.4.2 Input to Program UNDO

Input to program UNDO consists of the $UNDO namelist followed by any set of cards that is valid input to program CONVERT.

11.4.3 Output of Program UNDO

Output consists of a printed description of the diffraction corrected deck after it has been undone, and TAPE40.

TAPE40 is a disk file that contains the undone deck setup in a format that can be used by program CONVERT. Essentially, TAPE40 contains card images of Ho[N] and ACT[M] decks. Input to CONVERT usually consists of physical cards. To make CONVERT read TAPE40 instead of physical cards, the user must execute CONVERT with the command:

CONVERT, TAPE40.

instead of the usual command

CONVERT.

See example run in Appendix

11.4.4 The $UNDO Namelist

11.4.4.1 The $UNDO Namelist Variables And Their Types

DZZ	real array of length five
FZERO	real variable
ANGLE	real variable
GAMMA	real variable
ZZO	real variable
BETTAO	real variable
HASIS	logical variable of length five
PERIODIC	logical variable of length five
NCOMBS	integer variable

11.4.4.2 The $UNDO Namelist Variables And Their Defaults

DZZ	.5,.5,.5,.5,.5
FZERO	336.E6
ANGLE	0.
GAMMA	.378
ZZO	800.
BETTAO	50.
HASIS	F,F,F,F,F
PERIODIC	F,F,F,F,F
NCOMBS	2

11.4.4.3 The $UNDO Namelist Variables And Their Functions

DZZ:	Diffraction Correction Variable
FZERO:	Diffraction Correction Variable
ANGLE:	Diffraction Correction Variable
GAMMA:	Diffraction Correction Variable
ZZO:	Diffraction Correction Variable
BETTAO:	Diffraction Correction Variable
NCOMBS	The number of combs in the input card setup. There may be up to five combs in an input deck.
HASIS:	If HASIS(I) = .True., then there is no attempt to undo the diffraction correction of the Ho[N] (or Ho[NTAP]) deck describing the I'th comb.
PERIODC:	If PERIODC(I) = .True., then the ACT[M] or TCA[NTAP] deck describing the I'th comb would be deleted (see Sections 3.4.4 and 3.5.5 for a description of the defaults used when an ACT[M] or TCA[NTAP] deck is left out).

11.4.5 Example Deck Setup With Varying Namelist Parameter Examples

Consider the following input card setup:

```
$UNDO NCOMBS=4..........$
```

```
This is the CONVERT Title Card
```

```
$CONVERT NCOMBS=4, CODE=F,F,F,T....$
```

Ho[N] Deck

789

ACT[M] Deck

789

First Comb

Ho[N] Deck

789

789

Second Comb

789

ACT[M] Deck

789

Third Comb

Ho[NTAP]

789

TCA[NTAP]

789

Fourth Comb

A) First Comb

If HASIS(1) = .False., then the first Ho[N] deck has diffraction correction undone.

If HASIS(1) = .True., then the first Ho[N] deck is not changed. The deck is used as is.

If PERIODC(1) = .True., then the first ACT[M] deck is not used. A single 789 card on tape will replace the ACT[M] deck and the 789 card. That is, the deck setup for comb 1 will be changed to

Ho[N] Deck

789

789

B) Second Comb

If HASIS(2) = .False., then the second Ho[N] deck has diffraction correction undone.

If HASIS(2) = .True., then the second Ho[N] deck is not changed. The deck is used as is.

It does not matter whether PERIODC(2) = .True. or if PERIODC(2) = .False. since the second ACT[M] deck has already been deleted.

C) Third Comb

It does not matter if HASIS(3) = .True. or if HASIS(3) = .False. since the third Ho[N] deck does not exist. (A single 789 card represents the Ho[N] deck.)

If PERIODC(3) = .False., then the third ACT[M] deck is used.

PERIODC(3) must equal .False. since it takes at least one Ho[N] or ACT[M] deck to define a comb in RADC standard format.

D) Fourth Comb

If HASIS(4) = .FALSE., then the Ho[NTAP] deck has diffraction correction removed.

If PERIODC(4) = .False., then the fourth TCA[NTAP] deck is used.

If PERIODC(4) = .True., then the fourth TCA[NTAP] deck is not used. The fourth deck set will be changed to:

Ho[NTAP]

789

789

11.5 Program SAWPLOTS

11.5.1 Introduction

Program SAWPLOTS provides the capability of calculating the response of an ideal two comb saw device but does include diffraction effects. The diffraction may or may not be dependent on frequency (a user option).

The program uses two namelists and two RADC (either standard or coded) deck sets to provide the necessary information (electrode overlaps and positions) needed to calculate the response.

The program interfaces with either program GRAF or program GRAFTEK to plot the data as an insertion loss vs. frequency graph. Program GRAF (or GRAFTEK) can display the data as either relative or absolute insertion loss (see Sections 8.6 and 8.7). (See Appendix for example run.)

11.5.2 Input to Program SAWPLOTS

Input to program SAWPLOTS consists of the $INPUT title card, the $INPUT namelist, the specifications of two combs (using RADC standard or RADC coded format), the $FREQ title card, and the $FREQ namelist. The first 30 characters of the $INPUT title card will go on TAPE22 and be used by program GRAF to identify the plot. Thus, these 30 characters should contain the user's name and telephone extension.

11.5.2.1 The $INPUT Namelist

11.5.2.1.1 The $INPUT Namelist Variables And Their Type

FZERO:	Real
EPSTPR	Real
DVOVAV:	Real
VS:	Real
GGG:	Integer
ANGLE:	Real
GAMMA:	Real
ZMETERS:	Real
OVALAP	Real
CODE:	Logical array of length two
KOPT:	Integer
CFF:	Real
EPSO:	Real

11.5.2.1.2 The $INPUT Namelist Variables And Their Defaults

FZERO:	336.E6
EPSTPR:	4.55
DVOVAV:	.00058
VS:	3158.
GGG:	0

ANGLE:	0.
GAMMA:	.378
ZMETERS:	5.4E-3
OVALAP:	500.E-6
CODE:	F,F
KOPT:	100
CFF:	5.033885E-11
EPSO:	8.85E-12

11.5.2.1.3 The $INPUT Namelist Variables And Their Functions

FZERO:	(See Section 11.5.4)
EPSTPR:	(See Section 11.5.4)
DVOVAV:	(See Section 11.5.4)
VS:	(See Section 11.5.4)
GGG:	This variable tells the program which G function to use (see Section 11.5.4). If GGG=0, then the program uses Function G in its calculation. If GGG=3, for example, then function G3 is used. As of this writing only function G has been incorporated. Thus, GGG must equal zero.
ANGLE:	(See Section 11.5.4)
GAMMA:	(See Section 11.5.4)
OVALAP:	(See Section 11.5.4)
IN:	(See Section 11.5.4)
OUT:	(See Section 11.5.4)
CODE:	If CODE(I) = .False., then the I'th comb is specified in RADC standard format. If Code(I) = .True., then the I'th comb is specified in RADC coded format (I = 1 or 2).
KOPT:	If KOPT = 100 (default) then the diffraction will be frequency dependent. If KOPT = 0, then the diffraction will not be frequency dependent (see Section 11.5.4)
CFF:	Used by the G function.
EPSO:	Not currently used. Will probably be used in future G functions (i.e. G1, G2, G3...etc.).

11.5.2.2 Specifying The Two Combs

There are always two combs involved in a SAWPLOTS analysis. Each comb can be described using one of three possible format combinations. Thus, there are a total of nine possible format combinations for describing the two combs. The three possible format combinations for each

comb are as follows:

1. Ho[N] Deck
   789
   ACT[M] Deck
   789

2. Ho[N] Deck
   789
   789

3. Ho[NTAP] Deck
   789
   TCA[NTAP] Deck
   789

Mathematically, card setup 1 and card setup 2 will produce the same results no matter what ACT[M] deck is used in the first card setup (i.e. the ACT[M] deck is ignored). When the coded format is used (card setup 3) both the Ho[NTAP]and TCA[NTAP]decks must be present.

11.5.2.3 The $FREQ Namelist

11.5.2.3.1 The $FREQ Namelist Variables And Their Type

FINIT:	Real
FINC:	Real
IFREQ:	Integer

11.5.2.3.2 The $FREQ Namelist Variables And Their Defaults

FINIT:	321.E6
FINC:	20000
IFREQ:	1025

11.5.2.3.3 The $FREQ Namelist Variables And Their Functions

IFREQ: IFREQ gives the number of frequencies at which the insertion loss will be calculated.

FINIT: FINIT gives the starting frequency for which insertion losses will be calculated.

FINC: FINC gives the increment between the frequencies of consecutive insertion loss calculations.

Example:
IFREQ = 101
FINIT = 200.E6
F'NC = 1.E6

Insertion losses will be calculated at the following frequencies:

200.E6,201.E6,202.E6,203.E6......300.E6

11.5.3 Output From Program SAWPLOTS

11.5.3.1 Printed Output

At each frequency the following values will be printed out.

Frequency

$Y_1{}^2[f]$

$Y_2{}^2[f]$

[F,1]

[F,2]

[F,1] + [F,2]= Insertion Loss

The mathematical significance of these variables will be discussed in Section 11.5.4

11.5.3.2 TAPE22

Program SAWPLOTS produces TAPE22 as output. TAPE22 is usually a disk file that contains insertion loss vs. frequency data. TAPE22 can be used by either GRAF or GRAFTEK to produce an insertion loss vs. frequency graph.

11.5.4 Mathematical Specifications

11.5.4.1 Computing $Y_I(f)$

First program SAWPLOTS computes $Y_I(f)$ for I = 1, 2 and all f

from FINIT to [FINIT + (IFREQ-1)*FINC]

that is:

There are a total of "IFREQ" f's starting at frequency "FINIT" being incremented by frequency "FINC".

11.5.4.1.1 Case 1 - The I'th Comb is defined by an Ho[N] Deck (Code(I) = .False.)

For each f

$$Y_I(f) = \frac{1}{2\pi}\left(\sum_{N=-NMIN}^{NMAX} \frac{Ho[f,N]}{FZERO}\; e^{iN\pi}\; e^{i\xi[f,N]}\; e^{-i2\pi fT[N]}\right) \left(G\begin{bmatrix} GGG \\ F \\ FZERO \\ EPSTPR \\ DVOVAV \\ CFF \\ EPSO \end{bmatrix}\right)$$

The above equation is referred to as the "Basic Equation" for the CODE(I) = .False. case.

The variables in the above equation are defined as follows:

NMIN: The absolute value of the minimum N value of the Ho[N] deck defining the I'th comb.

NMAX: The maximum N value of the Ho[N] deck defining the I'th comb.

N: Goes from -NMIN to NMAX

FZERO: From the $INPUT Namelist

DVOVAV: From the $INPUT Namelist

F: Frequency as defined in Section 11.5.4.1

GGG: From the $INPUT Namelist.

EPSTPR: From the $INPUT Namelist.

CFF: From the $INPUT Namelist.

EPSO: From the $INPUT Namelist.

G: A function of f, FZERO, EPSTPR, DVOVAV, CFF, and EPSO. There can be more than one G function. Which G function is used depends on the value of GGG.

If GGG = 0 then subroutine G will be called.
If GGG = 1 then subroutine G1 will be called.
If GGG = 2 then subroutine G2 will be called, etc.

As of this writing, only subroutine G has been implemented (i.e. GGG must equal ZERO). Subroutines G, G1, G2, etc., return an imaginary number of the form C + Di.

For the current implementation of function G

$$C = \text{SIN}\left(\frac{\pi f}{2\ \text{FZERO}}\right) \quad (16{*}\text{DVOVAV}{*}\pi^2{*}\text{FZERO}^3{*}\text{OVALAP}{*}\text{CF})^{\frac{1}{2}}$$

D = 0.0.

T N : $$T[N] = \frac{N}{2(\text{FZERO})}$$

The two remaining quantities to be defined are:

ξ[F,N] and Ho[f,N]

If KOPT = 0 (KOPT from the $INPUT namelist), then the above quantities will be calculated as follows:

ξ[f,N] = 0

Ho[f,N] = Ho[N]

That is, Ho[f,N] equals the overlap value on the N'th Ho[N] card of the Ho[N] deck describing the I'th comb.

If KOPT = 100 (default) then the above quantities are calculated in the following way:

First, for each N, the following variables are computed:

<u>ANGLE</u> A constant from the $INPUT namelist.

<u>GAMMA</u> A constant from the $INPUT namelist.

<u>ZO[N,f]</u> $= f \frac{\text{ZMETERS}}{\text{VS}} + f \frac{N}{\text{2FZERO}}$

ZMETERS is from $INPUT
FZERO is from $INPUT
VS is from $INPUT

<u>BETAO[f,N]</u> $= f \frac{\text{Ho[N]} * \text{OVALAP}}{\text{VS}}$

OVALAP is from $INPUT
VS is from $INPUT
Ho[N] represents the OVALAP value of the N'th Ho[N]card of the Ho[N] deck describing the I'th comb.

<u>BETA[f,N]</u> = BETAO[f,N]

Now, call subroutine DIFF using the above 5 underlined variables as input.

Output from subroutine DIFF will be the following:

DL Ignored

APSLO This is ξ[f,N]

ALO This is Ho'[f,N]

For a given f let Q = the largest of all Ho' [f,N], N going from -NMIN to NMAX.

then

$$Ho[f,N] = \frac{Ho'[f,N]}{Q}$$

for all N from -NMIN to NMAX.

All the variables of the basic equation have now been defined. Calculation of the equation will be basd on the facts that

$$e^{iN\pi} e^{i\xi f,N} e^{i2\pi fT[N]}$$
$$= e^{(iN\pi \quad i\xi[f,N]+ i2\pi fT[N])}$$
$$= e^{i(N\pi + \xi[f,N] + 2\pi fT[N])}$$

In general $e^{i\theta} = \cos\theta + iSIN\theta$

Thus the above becomes

$$\cos(N\pi + \xi[f,N] + 2\pi fT[N]) + i\sin(N\pi + \xi(f,N) + 2\pi fT[N])$$

Thus the basic equation yields a complex number.

Let $A[f]$ = The real part of $Y_I[f]$

Let $B[f]$ = The imaginary part of $Y_I[f]$

Thus $A[f] + B[f]i$ represents the complex number $Y_I'(f)$.

Let $\hat{Q} = \sqrt{A[FZERO]^2 + B[FZERO]^2}$

(Note: A[f] and B[f] are calculated at specific increments of f. It is possible that the values A[FZERO] and B[FZERO] were not calculated at the exact frequency FZERO. In this case the values of A FZERO and B[FZERO]will be substituted by the calculations A[f] and B[f] for the value of f closest to FZERO.)

$$Y_I[f] = \sqrt{\frac{A[f]^2 + B[f]^2}{\hat{Q}}}$$

11.5.4.1.2 **CASE 2 - The I'th Comb is Defined by Ho[NTAP] and TCA[NTAP]decks. (CODE(I) = .True.)**

For each f

$$Y_I\,[f] = \frac{1}{2\pi} \sum_{NTAP=-NTAPMIN}^{NTAPMAX} \; \sum_{N=(NTAP)\;(IN+OUT)-K}^{(NTAP)\;(IN+OUT)+L} \frac{H_o[f,N]}{FZERO} \; e^{iN\pi} e^{i\xi[f,N]} e^{-2\pi f T[N]} \left(G \begin{bmatrix} GGG \\ f \\ FZERO \\ EPSTPR \\ DVOVAV \\ CFF \\ EPSO \end{bmatrix} \right)$$

The variables in the above equation are defined as follows:

NTAPMIN: The absolute value of the minimum NTAP value of the Ho[NTAP] deck defining the I'th comb.

NTAPMAX: The maximum NTAP value of the Ho[NTAP] deck defining the I'th comb.

NTAP: Goes from -NTAPMIN to NTAPMAX.

OUT: From the $INPUT Namelist.

IN: From the $INPUT Namelist.

K: K is defined as follows:

If "IN' is odd and NTAP = NTAPMIN, then K = 0

If "IN" is odd and NTAP >NTAPMIN, then $K = \frac{IN-1}{2}$

If "IN" is even the program will abort.

L: L is defined as follows:

If "IN" is odd and NTAP = NTAPMAX, then L = 0.

If "IN" is odd and NTAP <NTAPMAX, then $L = \frac{IN-1}{2}$

If "IN" is even the program will abort.

N: For each value of NTAP, N goes from:

(NTAP)(IN+OUT) - K

to

(NTAP)(IN+OUT) + L

GGG: From the $INPUT Namelist.

f: Frequency as defined in Section 11.5.4.1

FZERO: From the $INPUT Namelist.

EPSTPR: From the $INPUT Namelist.

DVOVAV: From the $INPUT Namelist.

CFF: From the $INPUT Namelist.

EPSO: From the $INPUT Namelist.

G: As described in Section 11.5.4.1.1.

T[N]: $$T[N] = \frac{TCA[NTAP]}{VS} + \frac{N-(NTAP)(IN+OUT)}{2[FZERO]}$$

"VS" is from the $INPUT namelist.

The two remaining quantities to be defined are:

$\xi[f,N]$ and $Ho[f,N]$

If KOPT = 0 (KOPT from the $INPUT namelist), then the above quantities will be calculated as follows:

$$\xi[f,N] = 0$$

$$Ho[f,N] = Ho[NTAP]$$

that is, Ho[f,N] equals the overlap value on the NTAP'th Ho[NTAP] card of the Ho[NTAP] deck describing the I'th comb.

If KOPT = 100 (default) then the above quantities are calculated in the following way:

First, for each N the following variables are computed:

<u>ANGLE</u> Directly from $INPUT

<u>GAMMA</u> Directly from $INPUT

$$\underline{ZO\,[N,f]} = f\,\frac{ZMETERS}{VS} + f\left(\frac{TCA\,[NTAP]}{VS} + \frac{N-(NTAP)\;(IN+OUT)}{2FZERO}\right)$$

ZMETERS is from $INPUT
FZERO is from $INPUT
VS is from $INPUT

$$\underline{BETAO\,[N,f]} = f\left(\frac{Ho^{D}\,[NTAP] \times OVALAP}{VS}\right)$$

OVALAP is from $INPUT

$$\underline{BETA}\,[N,f] = BETAO\,[N,f]$$

Now call subroutine DIFF using the above 5 underlined variables as input.

Output from subroutine DIFF will be the following:

DL	Ignored
APSLO	This is $\xi[f,N]$
ALO	This is $Ho'[f,N]$

For a given f let Q = the largest of all $Ho'[f,N]$ for all values of N

then

$$Ho\,[f,N] = \frac{Ho'[f,N]}{Q}$$

for all N

All the variables of the basic equation have now been defined. Calculation of the equation will be based on the facts that

$$e^{iN\pi}e^{i\,\xi[f,N]}e^{i2\pi fT\,[N]}$$

$$= e^{(iN\pi\, +\, i\,\xi[f,N]\, +\, i2\pi fT\,[N])}$$

$$= e^{i(N\pi\, +\,\xi[f,N]\, +\, 2\pi ft\,[N])}$$

In general $e^{i\theta} = \cos\theta + iSIN\theta$

Thus the above becomes

$$\cos(N\pi + \xi[f,N] + 2\pi fT\,[N]) + iSIN(N\pi + \xi[f,N] + 2\pi fT\,[N])$$

Thus the basic equation yields a complex number.

Let A [f] = The real part of $Y_I'(f)$.

Let B [f] = The *imaginary part of* $Y_I'(f)$.

Thus A [f] + B [f]i represents the complex number $Y_I'(f)$

$$\text{Let } \hat{Q} = \sqrt{A[FZERO]^2} + \sqrt{B[FZERO]^2}$$

NOTE: A [f] and B [f] are calculated at specific increments of f. It is possible that the values *A [FZERO] and B[FZERO] were not calculated at the* exact frequency FZERO. In this case the values of A [FZERO] and B [FZERO] will be substituted by the calculations A [f] and B [f] for the value of f closest to FZERO.)

$$Y_I[f] = \frac{\sqrt{A[f]^2 + B[f]^2}}{\hat{Q}}$$

11.5.4.2 <u>Further Calculations</u>

Program SAWPLOTS prints the following values at each frequency (GL = .02).

$Y_1(f)^2$

$Y_2(f)^2$

$$\text{Transducer Response [1]} = -10 \text{ Log} \left(\frac{2*GL*Y_1^2[f]}{(GL + Y_1^2[FZERO])^2} \right) = [F,1]$$

$$\text{Transducer Response [2]} = -10 \text{ Log} \left(\frac{2*GL*Y_2^2(f)}{(GL + Y_2^2[FZERO])^2} \right) = [F,2]$$

Total Insertion Loss = [F,1] + [F,2]

It is the Total Insertion Loss *that goes on* TAPE22 *to be plotted by* GRAF *or* GRAFTEK.

Note: The quantities when printed out are labeled as Insertion Loss [1] and Insertion Loss [2].

12.0 ELECTRON BEAM FABRICATION OF INTERDIGITAL TRANSDUCERS

Under previous contract to AMR, RADC/ET obtained hardware for fabricating SAW interdigital transducers by Electron Beam Lithography. This technique is considered to have excellent potential to yield transducers with linewidths (electrodes) as low as .2 micrometers. This is much finer than the minimum linewidths obtainable by the 10:1 optical projection technique, described in Section 5. As a result, SAW filters with higher center frequencies can be realized by E-BEAM.

Basically the steps involved in electron beam lithography are as follows. The surface of the SAW substrate, on which the transducer pattern is to be placed, is coated with a thin uniform layer of electron resist, such as polymethyl-methacrylate (PMM), or another polymer. A computer-controlled electron beam is passed over the resist-coated substrate in the areas in which the pattern is to appear. This beam is directed by programmed instructions from the user. As the electrons collide with the polymer molecules, the molecules are broken down, with the result that, after beam exposure, the average molecular weight of the polymer in a given area is inversely proportional to the charge (i.e. number of electrons) deposited by the beam in that area.

After beam exposure the substrate is placed in a special solution. The electron resist is soluable in this solution and the rate at which the molecules go into solution is inversely proportional to the molecular weight. Consequently, the more beam exposure an area of the resist has received, the faster that area is dissolved. The substrate is kept in the solution only long enough so that the areas irradiated by the beam are completely removed, thus exposing the SAW material in these areas, but the non-irradiated areas are still covered with the resist.

The entire sample is then coated with a thin layer of metal, using a vacuum deposition technique. It is then placed into another solvent to remove the remaining resist and the metal above the resist. The result is a SAW substrate with a metal (transducer) pattern on the surface, in the areas where the beam had passed. Therefore, in order to fabricate a given pattern, a user must provide the program instructions to guide the electron beam over the pattern areas at the proper rate.

ACSI's participation in this project was to develop a system of computer programs to generate the appropriate instructions to the electron beam pattern generator required to fabricate a given transducer pattern. This system requires as input only the physical description of the transducer, and produces as output a paper tape which can be read by the pattern generator to fabricate that transducer. All the intermediate steps, such as solving the electron optical equations, calculating the exposure due to cooperative exposure effects, and the determination of the beam scanning rates, are done by the system internally, thus enabling the design engineer, unfamiliar with electron beam lithography, to fabricate

SAW devices by this technique. Additionally, the paper tape, and a few machine settings (also calculated by the program) are all that is needed for an electron beam fabrication engineer to fabricate complex SAW devices.

12.1 Hardware Description

The electron beam fabrication system at RADC/ET, developed by Advanced Metals Research Corporation consists of three sub-systems: 1) an electron optical column and stage, 2) an analog electron optical control and display unit and 3) a digital control system. The sample is placed on the stage beneath the electron optical column, which is the source of the electron beam. Both the beam scanning rates and deflections and the stage movements are governed by the digital control system, which receives its directives from the user supplied paper tape. A pattern is built up by combinations of stage movements and beam deflections.

The substrate is positioned on a motorized stage with two normal axes, each of which is capable of locating the stage over a distance of 100 mm. The stage movements along each axis are independently monitored by a laser interferometer in 791 angstrom (.0791 m) steps. A 100 mm translational movement capability in 791 angstrom steps implies 1,264,222 individually accessible addresses on each axis. The x and y addresses of the stage are contained in two 21 bit binary words which are incremented or decremented by the interferometer as the stage is moved in response to commands from the digital control system.

The magnetic deflection field of the electron beam is also digitally addressed and contains $2^{14}-1$ addressable points in each of the two normal axes. The total size of the magnetic deflection field is adjustable by the operator and can be varied continuously from .75 mm to 2 mm on a side, depending on the resolution required.

The patterns within the scan field are sub-divided into rectangles. For each rectangle the paper tape supplies the digital controller with the x and y coordinates of two opposing corners of the rectangle, (each are represented by a 2^{14} bit binary word). When the rectangle has been defined by these coordinates (x_1, x_2, y_1, and y_2), the scanning sequence is initiated. The beam scans from (x_1, y1) to (x_2, y_1), then increments the y address and scans from (x_1, y_1+1) to (x_2, y_1+1). This procedure is repeated until the final scan for the rectangle, from (x_1, y_2) to (x_2, y_2), is completed. The rate at which the beam scans is also specified on the paper tape. At the end of each scanning sequence the digital controller looks for new instructions from the paper tape (these may be either stage movement or scanning instructions) until the end of the tape is reached.

12.1.2 Cooperative Exposure Effects

As mentioned earlier, the areas on the substrate which have absorbed charge doses greater than Q_{min} (see Figure 12.1) become the final metallized areas. To build up a pattern, we direct the electron beam so

that the areas we want metallized receive a dose greater than Q_{min} and the non-metallized areas receive less than Q_{min}. It may appear that the absorbed dose in a given area is simply the product of the electron beam current and the corresponding dwell time. However, the electrons are not all absorbed at the point at which they impinge on the slab, but rather have a distribution about this point. This is because the electrons scatter several times within both the resist and the substrate, and an electron may be several mean-free-paths away from the point at which it entered the resist before it has lost all its kinetic energy. Consequently, the effective absorbed charge in a given area is a certain fraction of the charge coming from the beam when directed at that area, plus a fraction of the beam charge when the beam is directed at other areas. Since the widths of the transducer electrodes are comparable to the electron mean-free-path, this cooperative exposure effect must be taken into account when calculating the dose rates for transducer electrodes.

In this section, we develop a cooperative exposure model for use in fabrication of interdigital transducers by electron beam lithography. Since it is the distribution of the absorbed charge which affects the final pattern, and since we can control only the deposited charge, (by directing the beam position and scanning rates), our model should tell us the deposited charge distribution which results in the desired absorbed charge distribution. For the case of interdigital transducers, we want the charge distribution along the length of the transducers to be such that it is greater than Q_{min} at the electrode sites and less than Q_{min} between the electrodes (see Figure 12.1). Silnce the electrode lengths are much greater than their widths we assume a constant charge distribution along the length dimension, and will vary only the distribution in the X (width) dimension.

We begin by making the following definitions:

x_i = The x-coordinate of the center of the i^{th} electrode.

$Q_{dep}(x)dx$ = the beam charge impinging on the resist, (i.e., the deposited charge), between x and x+dx.

$Q_{abs}(x)dx$ = The charge absorbed in the resist between x and x+dx.

$f(x' \rightarrow x)dxdx'$ = The fraction of the charge deposited between x and x+dx, which is absorbed between x' and x'+dx'. (This is the "Green's function" corresponding to the charge absorption).

The last definition allows us to relate the absorbed and deposited charge distributions by the following equation:

$$Q_{abs}(x) = \int_{-\infty}^{\infty} Q_{dep}(x')f(x' \rightarrow x)dx' \qquad (1)$$

Because of hardware constraints we are not free to vary the deposited charge distribution continuously, but only in discrete steps. Therefore, at this point, we impose the requirement that $Q_{dep}(x)$ takes the form (where a is the half width of each electrode):

$$Q_{dep}(x) = \begin{cases} Q_i & x_i - a < x < x_i + a \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where 2a is the "deposition width"; the width over which the beam is directed for each electrode. In terms of previously defined quantities, this is given by:

$$2a = N_{scans} \cdot d_B, \tag{3}$$

and:

Q_i = The areal charge deposited by the beam at the site of the i^{th} electrode. These are the quantities that we seek.

Now Equation (1) reduces to:

$$Q_{abs}(x) = \sum_{i=1}^{N} Q_i \int_{x_i-a}^{x_i+a} f(x' - x)dx', \tag{4}$$

for a transducer with N electrodes. Later, we will evaluate the integrals. We want the absorbed charge to be equal to Q_{min} at the finger edges, i.e.

$$Q_{abs}(x_j + a) = Q_{min}, \quad j = 1,2...N \tag{5}$$

Substituting this into Eq. (4)

$$\sum_{i=1}^{N} Q_i \int_{x_i-a}^{x_i+a} f(x' - x_j + a)dx' = Q_{min}, \quad j = 1,2...N \tag{6}$$

The above is a system of 2N equations in the N unknown Q_i's. We have more equations than unknowns since we have imposed the requirement that the deposition width be the same for each electrode. In the general case

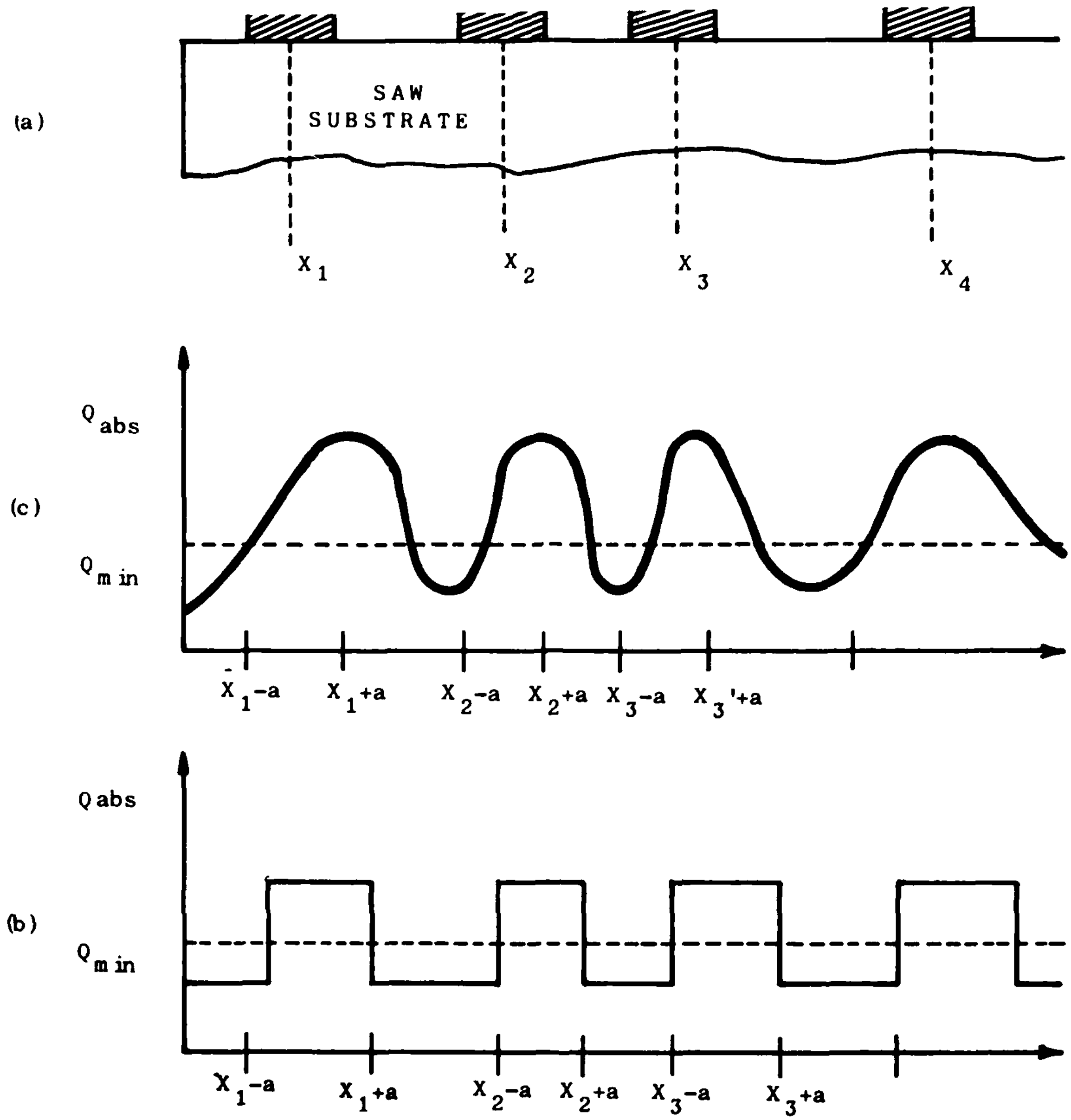


Figure 12.1 Charge Distribution required for Fabricating an Interdigital Transducer

(a) Position of transducer fingers on substrate.

(b) Ideal absorbed charge distribution required.

(c) Nominal absorbed charge distribution which will place the electrodes as shown.

we would have an additional N unknowns; the deposition width at each electrode. We are forced to choose the N boundary conditions we will satisfy out of the 2N required. We arbitrarily choose to meet the boundary condition on the left edge of each electrode, so we have the following in place of Equation (6):

$$\sum_{i=1}^{N} Q_i \int_{x_i-a}^{x_i+a} f\,(x' \rightarrow x_j - a)dx' = Q_{min},\ j = 1,2...N \quad (7)$$

which is a system of N equations in the N unknown Q_i's. This can be written in matrix form as:

$$F\vec{Q} = \vec{Q}_{min} \quad (8)$$

where the matrix elements are given by.

$$(F)_{ij} = \int_{x_i-a}^{x_i+a} f\ (x' \rightarrow x_j - a)dx', \quad (9)$$

$$(\vec{Q})_i = Q_i, \quad (10)$$

$$(\vec{Q}_{min})_i = Q_{min}, \qquad i,j = 1,2...N. \quad (11)$$

And the desired Q_i's are given by:

$$\vec{Q} = F^{-1}Q_{min}, \quad (12)$$

where F^{-1} is the inverse of the matrix F. (13)

We now turn to the evaluation of the integrals in Equation (9). At this point, f can be expressed approximately as the sum of two Gaussians, one representing primary and the other secondary backscatterers.

$$f(x' \rightarrow x) = c_1 e^{\frac{-(x'-x)^2}{B_1^2}} + C_2 e^{\frac{-(x'-x)^2}{B_2^2}} \quad (14)$$

where c_1, c_2, B_1 and B_2 are constants to be determined experimentally. Substituting this expression into Equation (9) gives:

$$(F)_{ij} = c_1 \int_{x_i-a}^{x_i+a} e^{-(x'-x_j)^2/B_1^2} dx' + c_2 \int_{x_i-a}^{x_i+a} e^{-(x'-x_j)^2/B_1^2} \tag{15}$$

Let $u_i = (x'-x_j)/B_1$ and $u_2 = (x'-x_j)/B_2$, then:

$$(F)_{ij} = B_1 c_1 \int_{(x_i-a-x_j)/B_1}^{(x_i+a-x_j)/B_1} e^{-u_1^2} du_1 + B_1 c_1 \int_{(x_i-a-x_j)/B_2}^{(x_i+a-x_j)/B_2} e^{u_2^2} du^2 \tag{16}$$

$$= \sqrt{\frac{\pi}{2}} B_1 c_1 \operatorname{erf}(\mu_1) \Bigg|_{\mu_1=(x_i-a-x_j)/B_1}^{\mu_1=(x_i+a-x_j)/B_1} + \sqrt{\frac{\pi}{2}} B_2 c_2 \operatorname{erf}(\mu_2) \Bigg|_{\mu_2=(x_i-a-x_j)/B_2}^{\mu_2=(x_i+a-x_j)/B_2} \tag{17}$$

where "erf" is the familiar "error function" defined by:

$$\operatorname{erf} z = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} dt \tag{18}$$

Substituting the limits:

$$(F)_{ij} = \sqrt{\frac{\pi}{2}} \left\{ B_1 c_1 \left[\operatorname{erf}\left(\frac{x_i+a-x_j}{B_1}\right) - \operatorname{erf}\left(\frac{x_i-a-x_j}{B_1}\right) \right] + B_2 c_2 \left[\operatorname{erf}\left(\frac{x_i+a-x_j}{B_2}\right) - \operatorname{erf}\left(\frac{x_i-a-x_j}{B_2}\right) \right] \right\} \tag{19}$$

which gives the elements of the matrix F in terms of known quantities. These are then used in Equation (12) to obtain the Q_i's.

Physically, the matrix element $(F)_{ij}$ represents the fraction of the beam charge deposited at the site of the j^{th} electrode, which is absorbed at the side of the i^{th} electrode. Usually, this fraction is very small for widely separated fingers so that the matrix F has only diagonal elements and a few off-diagonal elements. We can take advantage of this fact when inverting the matrix in order to reduce the required computer storage and computation time required.

12.2 Program Concepts

The purpose of the computer program is to generate paper tapes containing the appropriate instructions to the pattern generator for fabricating interdigital transducer patterns. It should require as input only the physical dimensions of the transducers, plus a few constants describing the SAW material and the resist. The output should include the paper tape itself as well as brief meaningful instructions to the operator, and other printed output describing the various dimensions of the pattern. By keeping the input, output and operating instructions as simple as possible, all the complexities in determining the appropriate exposures will be contained within the program modules, hidden from view, thus reducing the amount of interpretation required from either the device designer (program user) or the device fabricator (pattern generator operator).

12.2.1 Program Logic

Figure 12.2 illustrates an example of a typical transducer pattern to be fabricated by the electron beam lithographic technique, and defines some of the dimensions used to locate the devices on the slab. Recall that the 10X masters described in the previous section contained either a single transducer pattern or all the transducers comprising a single SAW device (or "SAW filter"). However, Figure 12.2 shows that the SAW substrates used in the electron beam lithographic technique may contain several SAW filters. In some cases, a certain SAW filter will simply be repeated several times on the slab so that the reproducibility can be tested. In other cases, each SAW filter will be different and all the filters on the slab will comprise what is known as a "filter bank". Each device on the slab is called a "channel". (In this context a "device" means the combination of two or more interdigital SAW transducers aligned along the same acoustic path.)

The physical dimensions, placement and overlap of the electrodes in each of the transducers on the slab are described in the usual way by means of the standard RADC/ET format and/or the coded RADC/ET format (see Section 3). Consequently, program CONVERT is required to convert the data from these formats to physical dimensions in the Raytheon format (see Section 3) before running AMR, the Electron Beam Fabrication Program. This is also necessary when running the 10X master fabrication programs, ELECTRO and HUGHES, described in Sections 5.0 and 6.0. The flow of the logic in program AMR is illustrated in Figure 12.3 Each step is described in more detail below.

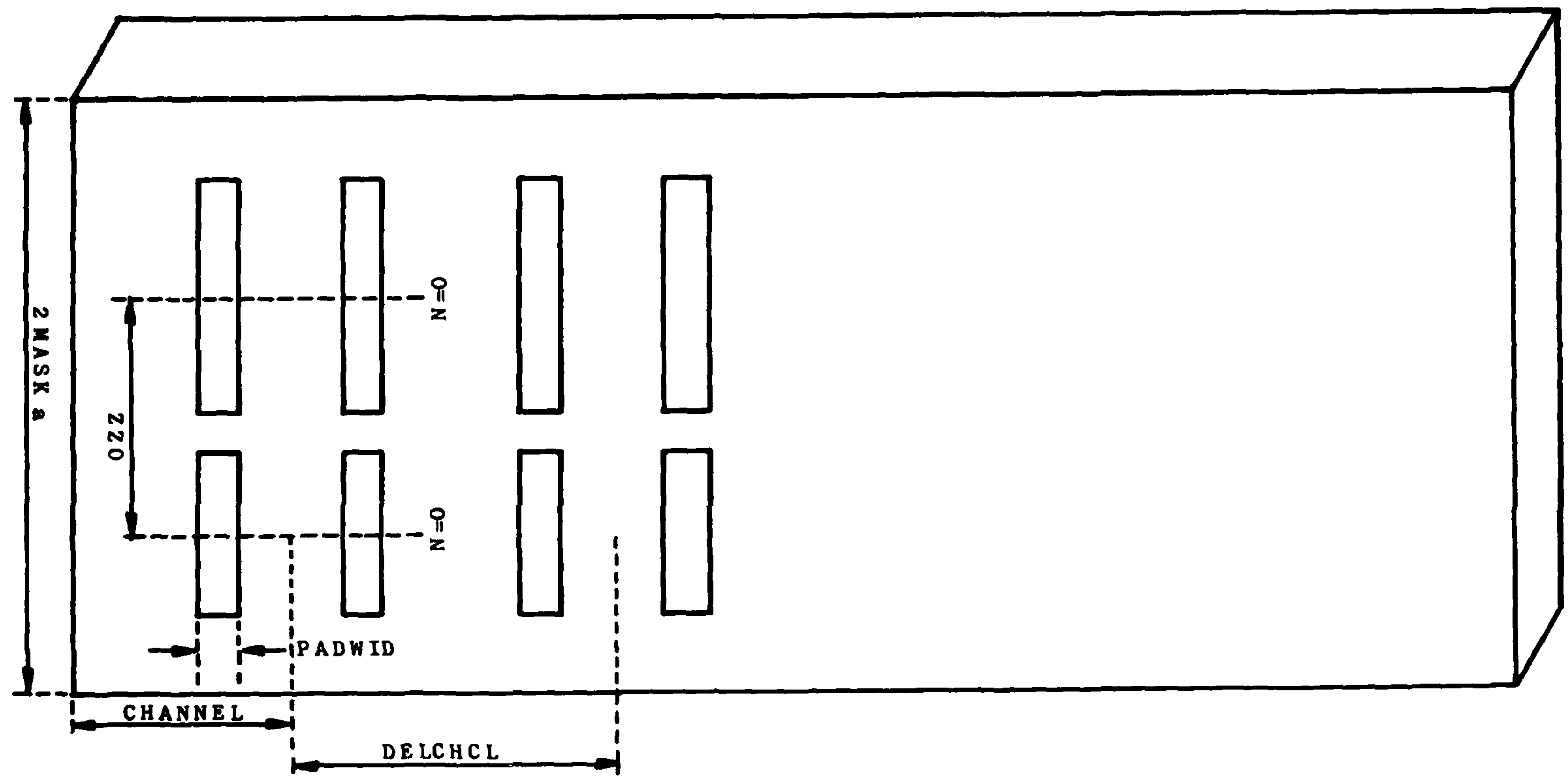


Figure 12.2 Orientation of Transducers on SAW Substrate When Fabricated by Electron Beam Lithography

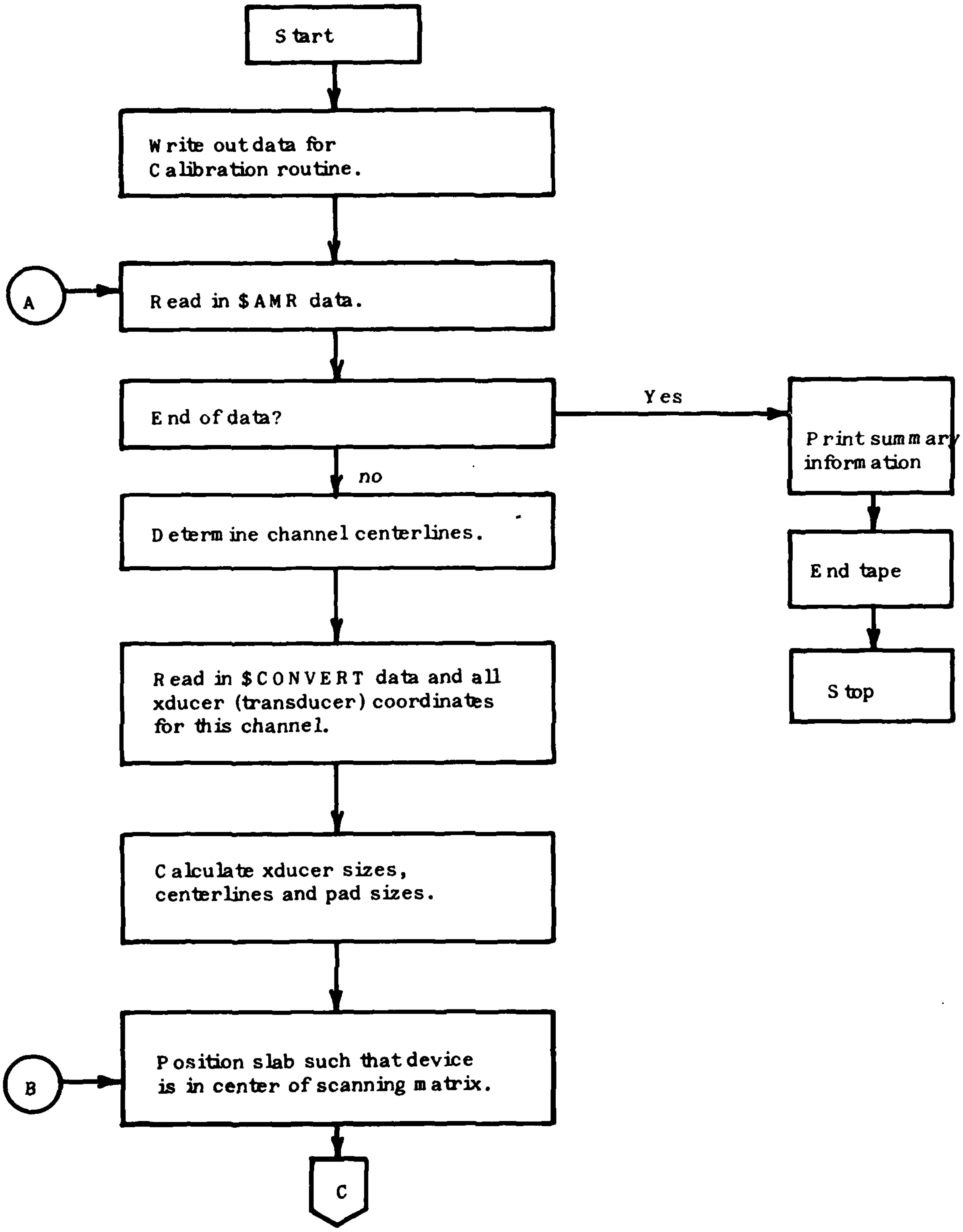


Figure 12.3 Flow Diagram of AMR Program Logic

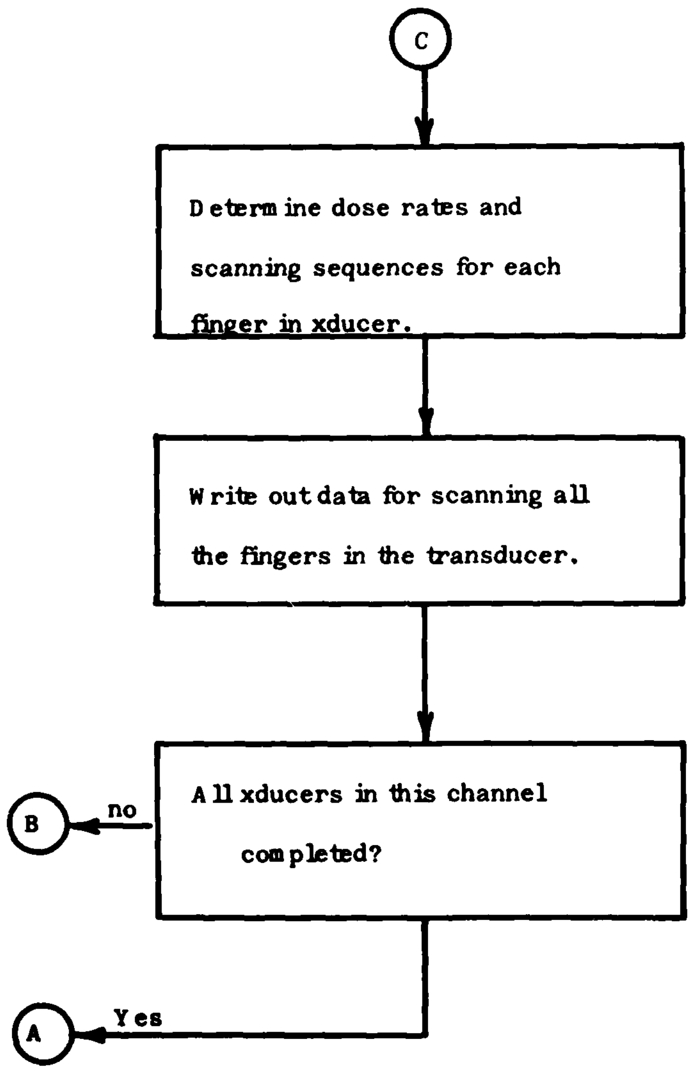


Figure 12.3 Flow Diagram of AMR Program Logic (cont.)

Before the electron beam fabricator can be used to generate SAW patterns, the operator must perform a series of checks on the slab alignment, electron optics and other system variables. To assist in these checks, the first few blocks of every paper tape program contain what is called a "calibration routine". This routine is simply a series of scanning and stage motion commands, and is the same regardless of the pattern to be created. Execution of program AMR always begins with these commands being punched on paper tape.

The program then reads in the data from the $AMR namelist. This information is used to locate the various transducers on the slab. Execution of the program is repeated (except for writing out the calibration routine) until there are no more $AMR cards on the input file. Usually there will be either one $AMR card for the entire filter bank, or one for each channel in the filter bank, although intermediate combinations are also allowed.

When all $AMR cards have been processed, the paper tape will then contain all the instructions required for each channel on the slab. The program then writes out a few blank records on the paper tape file to insure that all the buffers are empty at program termination. Finally the program prints out a few numbers for use by the operator when running the paper tape. These include the size of the scanning field, the beam current, and the width at which the electron aperture is to be set.

Once the $AMR data has been read in and the location of the transducers on the slab has been determined, the program reads in the data generated by program CONVERT (see Section 3). These data describe the physical dimensions of each transducer in the channel. The transducer electrode dimensions and positions (from file TAPE9) along with the $CONVERT variables (read from TAPE28) are all that are necessary to describe the transducer uniquely. These data are then used to calculate the transducer lengths and the corresponding pad lengths. The transducer centerlines, with respect to the slab coordinate system, are also calculated.

The size of the scanning field, d_f, is then determined from the size of the largest pattern. The available field sizes presently are .7 mm to 2 mm on a side in .1 mm increments. The program selects the smallest of these which will contain the largest transducer pattern. Since there are 2^{14} divisions on each side of the square field the required beam width is given by:

$$d_B = d_F/16383.$$

Thus the areas covered by adjacent passes of a beam of this width will intersect, insuring that all the area contained in a scanning rectangle will be covered by the beam.

The electron optical equations are then solved for the beam current and electron aperture size required for a beam of this diameter. Before the program terminates, these parameters are printed out for later use by the fabricator operator, when setting up the system before running through the paper tape. (The electron optical equations used by this program are developed in detail in Section 12.2.2.)

Because of hardware limitations, the assumption must be made that all transducer electrodes are the same width, w. (however, this is usually the case anyway.) It has been determined experimently on the RADC/ET electron beam fabricator that the width of a line fabricated using a single pass of the beam is typically 3 to 5 times the diameter of the beam, depending on exposure. The line width is greater than the beam diameter because electron scattering in the resist and substrate cause a distribution of beam charge about the point of beam impact. We define the minimum line width as:

$$W_{min} = 3d_B,$$

and require N_{scans} adjacent passes of the beam to fabricate a line of width W, where N_{scans} is given by:

$$N_{scans} = \frac{W}{W_{min}},$$

truncated to an integer. The restriction that every transducer electrode on the slab is to be the same width also restricts the variable N_{scans} to be a constant as well.

We have restricted the size of the area to be covered by the electron beam for fabricating each electrode. However,the beam exposure can still be varied, thus varying the width of the developed line. The exposure is controlled by varying the rate at which the beam passes over a given area. The scanning rates and corresponding dwell times, which can be specified by a paper tape program and used as input. The scanning rate of the beam can only be varied in factors of 2, but the software (i.e. the paper tape program) can circumvent this restriction. Since the exposure is cumulative, and directly proportional to the beam dwell time, an arbitrary exposure in a given area can be achieved by multiple passes of the beam over the area at different rates. The final exposure is the sum of the exposures of the individual scans. This procedure is called "redundant scanning" and is incorporated into the FINGER module.

Next the program determines the scanning sequence (exposed areas and rates) required for each transducer pattern. Since the absorbed charge in a given area is equal to a fraction of the charge deposited by the beam in that area, plus a fraction of the charge deposited in neighboring areas (called the "cooperative exposure" effect), determination of the scanning sequence required to fabricate a given pattern is a formidable task. The cooperative exposure model developed for this program is described in Section 12.1.2. However, since many of the constant parameters are still to be determined experimentally, this model was never incorporated into the program. As it currently stands the program directs the beam to pass through each area at the same rate. The model described in Section 12.2.2 is available to incorporate into the program when the constants have been determined.

Once the scanning sequences have been determined, the appropriate scanning instructions are punched onto the paper tape. The transducer pads patterns are exposed next, by scanning the beam over a single rectangle in the appropriate area, above and below the transducer electrodes. When both the electrodes and pads have been exposed, the stage is moved in the X direction to the center of the transducer pattern for the next transducer in the channel, and the above procedure is repeated for the next transducer. When all transducer patterns in a given channel have been exposed, the logic returns to read another $AMR card.

12.2.2 Electron Optics

Once the desired width of the electron beam has been determined it is necessary to calculate the electron gun conditions which will result in a

beam of this diameter. These conditions are determined from a solution to the electron optical equations. What is meant by the "beam width" is actually the full width at half maximum of a Gaussian distribution of electrons.

The beam diameter d_B is given from:

$$d_B^2 = d_g^2 + d_f^2 + d_c^2, + d_s^2, \tag{20}$$

where the quantities on the right represent the contributions to the beam spreading due to geometric and chromatic and spherical aberration effects. These terms are given by:

$$d_g^2 = \frac{4 i_B}{\pi^2 \beta \alpha^2}, \tag{21}$$

$$d_f = .61\lambda/\alpha, \tag{22}$$

$$d_c^2 = C_c^2 \left(\frac{\Delta i}{i} + \frac{\Delta E}{E}\right)^2 \alpha^2 \equiv Q\alpha^2 \tag{23}$$

$$d_s^2 = (3C_s)^2 \alpha^6 \equiv R\alpha^6 \tag{24}$$

where i_B is the beam current and α is the semi-angle of the beam point subtended at the aperture. The remaining quantities are constants; β is the beam brightness, λ the electron wavelength, $\Delta i/i$ and $\Delta E/E$ are the fractional instabilities in the beam current and energy, respectively and C_c and C_s are respectively the chromatic and spherical aberration coefficients. For the RADC/ET system the constants are:

$$\begin{aligned}
\beta &= 10^9 \text{ amp}/M^2.\text{sr} \\
\lambda &= \left(\frac{2X10^{-18} M^2.\text{ev}}{E}\right)^{1/2} = .1X10^{-10} M \\
C_c &= .06M \\
C_s &= 3.5M \\
\Delta i/i &= 10^{-5} \\
\Delta E/E &= 10^{-4} \\
E &= 20 \text{ X } 10^3 \text{ eV}
\end{aligned} \tag{25}$$

With this information we have an equation relating the beam current and the aperture size, for a given beam diameter. Thus, according to this equation, there exists an infinite number of combinations of α and i_B to obtain a desired beam diameter. The approach we take is to choose the solution which gives the highest beam current, since high electron fluxes will decrease the fabrication time required.

Equation 26 can be written as:

$$d_B^2 = \frac{4i_B}{\pi^2 \beta\alpha^2} + \frac{(.61\lambda)^2}{\alpha^2} + Q\alpha^2 + R\alpha^6. \tag{26}$$

Solving for i_B:

$$i_B = \frac{\pi^2\beta}{4}\left[d_B^2\ \alpha^2 - (.61\lambda)^2 - Q\alpha^4 - R\alpha^8\right] \tag{27}$$

The maximum beam current is obtained from a solution to:

$$\frac{\partial i_B}{\partial \alpha} = 0 = \frac{\pi^2 \beta}{4} \left[d_B^{\ 2}\ 2\alpha - Q4\alpha^3 - R8\alpha^7 \right]$$

$$\Rightarrow \alpha^6 + \frac{Q}{2R}\ \alpha^2 - \frac{d_B}{4R} \doteq 0, \tag{28}$$

which is a cubic equation in α^2. Eq. (28) specifies the aperture angle at which the solution to Eq. (27) gives the maximum beam current. Unfortunately, we are not free to vary continuously. Utilizing the available aperture sizes, d, we can relate α to d by:

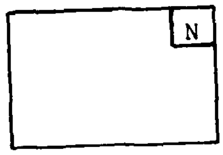
$$\sin \alpha = \frac{d/2}{\ell} \tag{29}$$

where $\ell = 70.4X10^{-3}$m on the RADC/ET system. So we first solve Eq. (28) for the optimum α, and then the corresponding aperture size d, is determined from Eq. (29). We then chose the closest available aperture size. This value of d is then used with Eqs. (29) and (27) to obtain the required beam current.

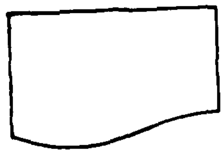
Program AMR contains a module which solves these equations and prints out the aperture size and the beam current. These quantities are then transmitted to the SEM operator, along with the paper tape.

APPENDICES

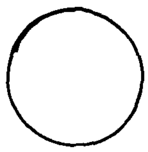
The following conventions are used throughout the Appendices.



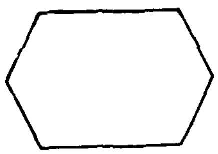
The running of a program. The N'th program to run.



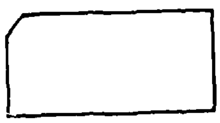
Printed Output



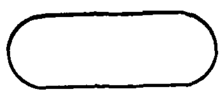
Disk file acting like magnetic tape



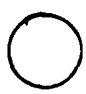
Physical Magnetic Tape



Punched Card Input



Plots



Flow Chart Connector

APPENDIX A

GILPM Run Deck and Data Flow Chart

The following run deck was used to simulate the example described in Section 9.4. These programs are collectively known as RC#2M. The flow chart presents the data flow diagram of the preceding run.

```
ELTER,CM202000,T200.                                      2562    ELTERMAN
PURGE,A,TAPE22,ID=COLVIN.
EXIT(U)
REQUEST,TAPE22,*PF.
ATTACH,RIND,RIND,ID=COLVIN,MR=1.
ATTACH,GILP,GILPM,ID=COLVIN,MR=1.
ATTACH,MERGE,MERGEGX3716,ID=SLOBOD,MR=1.
ATTACH,ICAMG,ICAMGX3716,ID=SLOBOD,MR=1.
ATTACH,DECODE,DECODGX3716,ID=SLOBOD,MR=1.
ATTACH,MATCH3,MATCH3,ID=SLOBOD,MR=1.
ATTACH,SEE,SEEIT,ID=COLVIN,MR=1.
ATTACH,GRAF,GRAF,ID=SLOBOD,MR=1.
ATTACH,PEN,ONLINEPEN.
LIBRARY,PEN.
REQUEST,PLOT,*Q.
DISPOSE,PLOT,*PL.
ATTACH,TAPE11,MATRIX449495MHZ,ID=COLVIN,MR=1.
ATTACH,TAPE12,MATRIX532495MHZLCFFR,ID=COLVIN,MR=1.
SEE,TAPE11.
SEE,TAPE12.
LDSET,PRESET=ZERO.
MERGE.
RETURN,TAPE11.
RETURN,TAPE12.
LDSET,PRESET=ZERO.
ICAMG.
RFL,60000.
LDSET,PRESET=ZERO.
DECODE.
SEE,TAPE10.
REWIND,TAPE10.
RFL,130000.
LDSET,PRESET=ZERO.
MATCH3.
FTN,SL.
RFL,177000.
LDSET,PRESET=ZERO.
RETURN,TAPE11.
RETURN,TAPE12.
ATTACH,TAPE11,MATRIX449495MHZ,ID=COLVIN,MR=1.
ATTACH,TAPE12,MATRIX532495MHZLCFFR,ID=COLVIN,MR=1.
RIND.
LOAD(GILP)
LOAD(LGO)
EXECUTE.
REWIND,TAPE22.
REWIND,TAPE24.
COPYBF,TAPE24,TAPE22.
REWIND,TAPE22.
CATALOG,TAPE22,TAPE22,ID=COLVIN,MR=1.
RFL,50000.                                                               0002
LDSET,PRESET=ZERO.                                                       0002
GRAF.                                                                    0002
7897897897897897897897897897897897897897897897897897897897897897897897897897
 $MERGE NFREQ=1024 $
7897897897897897897897897897897897897897897897897897897897897897897897897897
     2
STOP
STOP
          50           0           50           0
RIND        375.0       940.00      1000.00      30.0          235
```

```
STOP
RING       615.0      665.00     1000.00    30.0
STOP
SERIES          2
    1  1
    1  2
STOP
STOP
PARALLEL
STOP
        50          0
 TWO FILTERS IN PARALLEL  427/547 MHZ
        427         547
 1024
789789789789789789789789789789789789789789789789789789789789789789
 $DECODE NFREQ=1024, NFIL=2 $
789789789789789789789789789789789789789789789789789789789789789789
    COLVIN PLOTS EXT 3585
 $MATCH SCOPE=F, LABEL=T, GRAPH(2)=T, CHART=T, JUNIT=11 $
 FIRST CHART INPUT WITH MATCHING.  SECOND CHART MEANINGLESS.  RC#2.S
 $ZLIST IREP=2 $
789789789789789789789789789789789789789789789789789789789789789789
      SUBROUTINE ILCALC(IL,ICAP,REZT)
      REAL IL,ICAP(8),REZT(8)
      TE=100.*(ICAP(1)*REZT(1) + ICAP(2)*REZT(2))
      IL=-10.*ALOG10(TE*TE)
      RETURN
      END
789789789789789789789789789789789789789789789789789789789789789789
 $QIND Q=30.,30., SRF=1000.E6,1000.E6, L=400.E-9,172.E-9 $
789789789789789789789789789789789789789789789789789789789789789789
 $SIL PRNT=T,C=.2E-12,.2E-12,L=400.0E-9,172.0E-9,R=13.0,15.8 $
 4
+SNRG+SMLC+SNQ1+SMQ2+SIC1
-SIC1
+SNRG+SMLC

-SIC1
+SIC1+SNP1+SNA1+SNB1


+SNRG+SMLC

+SNRG+SMLC+SNQ3+SMQ4+SIC2
-SIC2


-SIC2
+SIC2+SNP2+SNA2+SNB2
 4
+SNRG+SMLO+SNQ1+SMQ2+SIC1
+SNVG
+SNRG+SMLO

-SIC1



+SNRG+SMLC
+SNVG
+SNRG+SMLC+SNQ3+SMQ4+SIC2
-SIC2


-SIC2
+SIC2+SNP2+SNA2+SNB2
```

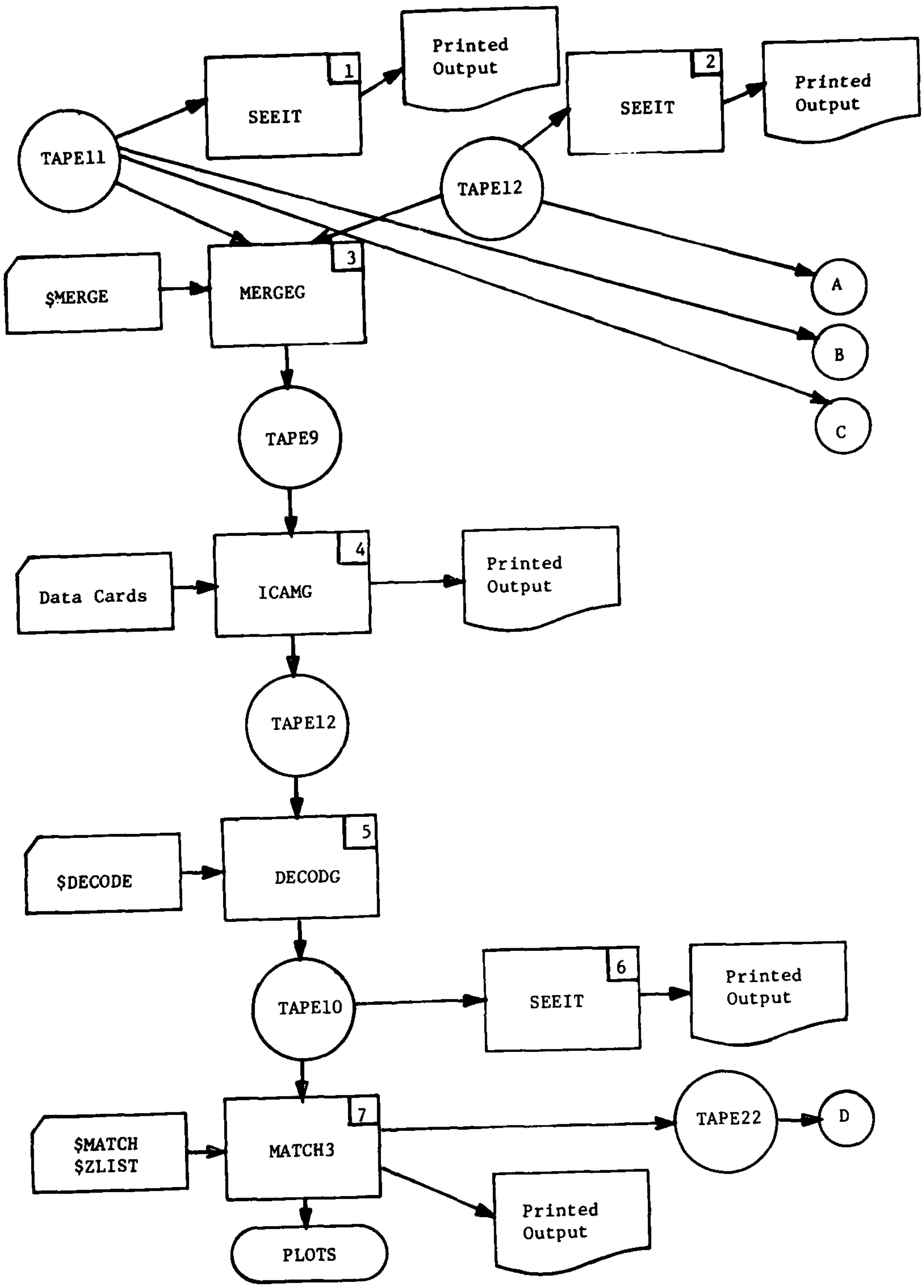
Description of first four by four matrix
(Matrix N_1 in example)

The four preceding the description indicates a four by four matrix will follow. The blank lines indicate a zero entry.

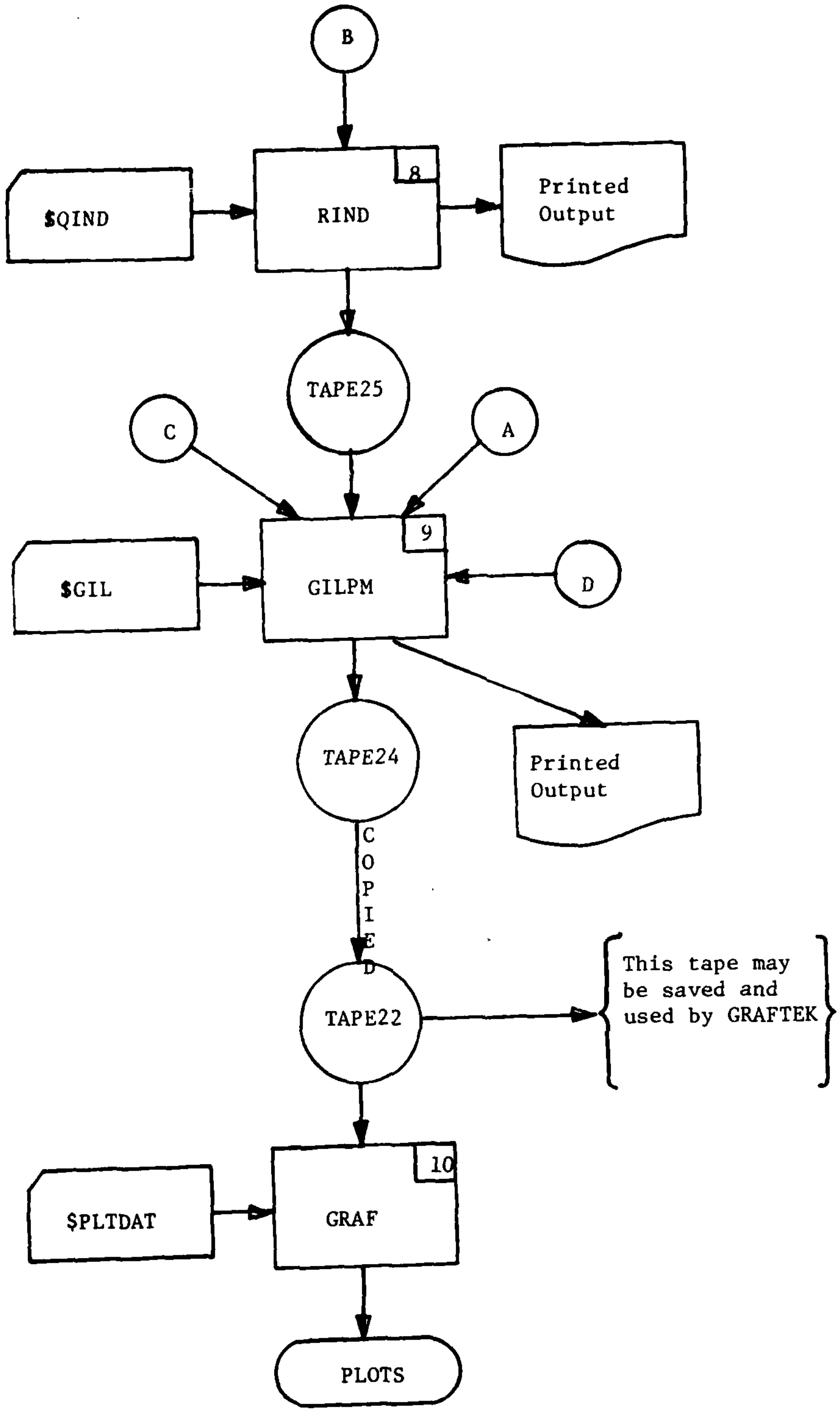
Description of second four by four matrix.
(Matrix N_2 in example)

```
4
+SNRG+SMLO+SNO1+SMO2+SIC1
-SIC1
+SNRG+SMLC
+SNVG
-SIC1
+SIC1+SNR1+SNA1+SNB1

+SNRG+SMLC

+SNRG+SMLO+SNO3+SMO4+SIC2
+SNVG

-SIC2
789789789789789789789789789789789789789789789789789789789789789789789789789789
 $PLTDAT FZERO=495.0E6,495.0E6,DIV=24.0E6,48.0E6,UNCAL=99.0,99.0 $
678967896789678967896789678967896789678967896789678967896789678967896789678967896789
```

Description of third four by four matrix. (Matrix D in example)



The following is the data flow diagram of the GILPM Run.


B
8
$QIND
RIND
Printed
Output
TAPE25
C
A
9
$GIL
GILPM
D
TAPE24
Printed
Output
COPIED
TAPE22
This tape may
be saved and
used by GRAFTEK
10
$PLTDAT
GRAF
PLOTS

APPENDIX D

The following is the version of program RIND that was compiled for use in the preceding sample run.

```
      PROGRAM RINO(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,TAPE11,TAPE25)
      REAL L
      COMPLEX Z1,Z2,Z3
      DIMENSION QQ(8)
      DIMENSION T(10),K(8)
      DIMENSION L(4),Q(4),SRF(4)
      NAMELIST /QIND/ L,SRF,Q
      REWIND 11
      REWIND 25
      READ(11,1) T,K
    1 FORMAT(10A10,8I2)
      DO 50 J=1,8
   50 QQ(J)=0.
      TWOPI=6.283185308
      DO 51 J=1,4
      L(J)=10.E-9
      SRF(J)=1.E9
      Q(J)=100.
   51 CONTINUE
      READ(5,QIND)
      WRITE(6,QIND)
   10 READ(11,20) F,Z1,Z2,Z3
   20 FORMAT(6020)
      IF(EOF(11).NE.0) GO TO 30
      DO 60 I=1,2
      R=(TWOPI*F*L(I))/Q(I)
      C=1./(L(I)*((TWOPI*SRF(I))**2 +(R/L(I))**2))
      D=((TWOPI*F*R)/L(I))**2+((1./(L(I)*C))-(TWOPI*F)**2)**2
      QQ(2*I-1)=(R/((L(I)*C)**2))/D
      QQ(2*I)=((1./C)*((1./(L(I)*C))-(TWOPI*F)**2-(R/L(I))**2))/D
   60 CONTINUE
      WRITE(25,35) (QQ(J),J=1,8)
   35 FORMAT(6020)
      GO TO 10
   30 REWIND 11
      REWIND 25
      WRITE(6,90)
   90 FORMAT(6X,9HFREQUENCY,12X,2HQ1,12X,2HQ2,12X,2HQ3,12X,2HQ4,
     112X,2HQ5,12X,2HQ6,12X,2HQ7,12X,2HQ8)
      READ(11,1) T,K
  100 READ(11,20) F,Z1,Z2,Z3
      READ(25,35) (QQ(J),J=1,8)
      IF(EOF(11).NE.0) GO TO 200
      IF(EOF(25).NE.0) GO TO 200
      WRITE(6,110) F,(QQ(J),J=1,8)
  110 FORMAT(1X,9E14.4)
      GO TO 100
  200 REWIND 11
      REWIND 25
      END
```

The following is an example of a GETMATD run.

```
ELTER,CM74000,T10.                                           2582    ELTERMAN
FTN,SL.
LGO.
7897897897897897897897897897897897897897897897897897897897897897897897897897897
      PROGRAM SYNTXCK(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      INTEGER SIZE,M(16,16,64)
      READ(5,1) N
1     FORMAT(I1)
      DO 10 J=1,N.
      CALL GETMATD(M,SIZE)
10    CONTINUE
      END
      SUBROUTINE GETMATD(M,SIZE)                                              001
      INTEGER M(16,16,64),SIZE,TEMP(64),A,B,C,D                               001


      READ(5,1) SIZE                                                          001
1     FORMAT(I2)                                                              001
      WRITE(6,15)
15    FORMAT(1H1)
      WRITE(6,99)SIZE                                                         001
      IF(SIZE.LT.1.OR.SIZE.GT.16)STOP "ILLEGAL MATRIX SIZE"                   001
99    FORMAT(1X,*MATRIX OF SIZE *,I2,* SQUARED*)                              001
      KSIZE=SIZE                                                              001
      DO 10 I=1,SIZE                                                          001
      DO 20 J=1,KSIZE                                                         001
      READ(5,2) (M(I,J,K),K=1,64)                                             001
2     FORMAT(16(A1,A2,A1,A1))                                                 001
                                                                              001
      DO 30 L=1,64                                                            001
      TEMP(L)=M(I,J,L)                                                        001
30    CONTINUE                                                                001
                                                                              001

      DO 40 L=2,62,4                                                          001
      TEMP(L)=3H                                                              001
      IF(M(I,J,L).EQ.2HSM) TEMP(L)=3H  S
      IF(M(I,J,L).EQ.2HSI) TEMP(L)=3H I/S
40    CONTINUE                                                                00:
      IF(M(I,J,1).EQ.1H )TEMP(1)=1H
      WRITE(6,3) (TEMP(L),L=1,64)                                             00:
3     FORMAT(1H0,*S1J*      *,16(A1,A3,A1,A1,2X))                             00:
                                                                              00:
C     NOW TO CHECK SYNTAX                                                     00
                                                                              00
      DO 50 L=1,61,4                                                          00
      A=M(I,J,L)                                                              00
      B=M(I,J,L+1)                                                            00
      C=M(I,J,L+2)                                                            00
      D=M(I,J,L+3)                                                            00
      IF(A.EQ.1H ) GO TO 75                                                   00
      IF(B.EQ.2HSI.AND.C.EQ.1HQ)GO TO 100
      IF(B.NE.2HSN.AND.(C.EQ.1HA.OR.C.EQ.1HB)) GO TO 100                      00
      IF(A.NE.1H-.AND.A.NE.1H+)GO TO 100                                      00
      IF((B.NE.2HSM.AND.B.NE.2HSN).AND.B.NE.2HSI) GO TO 100                   00
      IF(C.EQ.1HL.AND.D.EQ.1HQ)GO TO 50                                       00
      IF(C.EQ.1HR.AND.D.EQ.1HG) GO TO 50                                      00
      IF(C.EQ.1HV.AND.D.EQ.1HG) GO TO 50                                      00
                                                                              00
```

```
      IF(((D.EQ.1H1.OR.D.EQ.1H2).OR.(D.EQ.1H3.OR.D.EQ.1H4)).OR.            001
     1((D.EQ.1H5.OR.D.EQ.1H6).OR.(D.EQ.1H7.OR.D.EQ.1H8))) GO TO 70          001
      GO TO 100                                                           002
70    IF(((((C.NE.1HC.AND.C.NE.1HR).AND.C.NE.1HL).AND.C.NE.1HA)
     1.AND.C.NE.1HB).AND.C.NE.1HQ) GO TO 100
      GO TO 50                                                            002
75    LL=L+1                                                              002
      DO 80 LM=LL,64                                                      002
      IF(M(I,J,LM).NE.1H .AND.M(I,J,LM).NE.2H  ) GO TO 100                002
80    CONTINUE                                                            002
      GO TO 20                                                            002
50    CONTINUE                                                            002
20    CONTINUE                                                            002
10    CONTINUE                                                            002
      WRITE(6,4)                                                          002
4     FORMAT(1H0,*  NO SYNTAX ERRORS*)                                    002
      WRITE(6,15)
      RETURN                                                              002
                                                                          002

100   LT=(L+3)/4
      WRITE(6,5)                                                          002
5     FORMAT(1H0,* A SYNTAX ERROR HAS OCCURRED,EXECUTION WILL BE TERMINA002
     1TED.*)                                                              002
      WRITE(6,6) LT
6     FORMAT(1H0,* AN ERROR HAS OCCURRED IN FIELD NUMBER*,I3)
      WRITE(6,7)                                                          002
7     FORMAT(1H0,* THE FOLLOWING IS THE FAULTY CARD IMAGE*)               002
      WRITE(6,8) (M(I,J,KK),KK=1,64)                                      002
8     FORMAT(1H0,10X,16(A1,A2,A1,A1))                                     002
      WRITE(6,9)                                                          002
9     FORMAT(1X,10X,8(*1234567890*))                                      002
      STOP "BAD MATRIX DESCRIPTION SYNTAX"                                002
      END                                                                 002
789789789789789789789789789789789789789789789789789789789789789789789789789789
1
 4
+SNRG+SMLO+SMLI+SIC1
-SIC1
+SNRG+SMLO

-SIC1
+SIC1+SNR1+SNA1+SNB1

+SNRG+SMLO
+SMLO-SNB2+SMB3+SNRG
+SNRG+SMLO+SMLZ+SIC2
-SIC2

-SIC2
+SIC2+SNR2+SNA2+SNB2
789789789789789789789789789789789789789789789789789789789789789789789789789789
789789789789789789789789789789789789789789789789789789789789789789789789789789
 15.01.31.MFA. AB      SCOPE 3.4.4        * CDC566A  A.F.G.L.
678967896789678967896789678967896789678967896789678967896789678967896789678967
```

The number of matrices
to be syntactically checked.

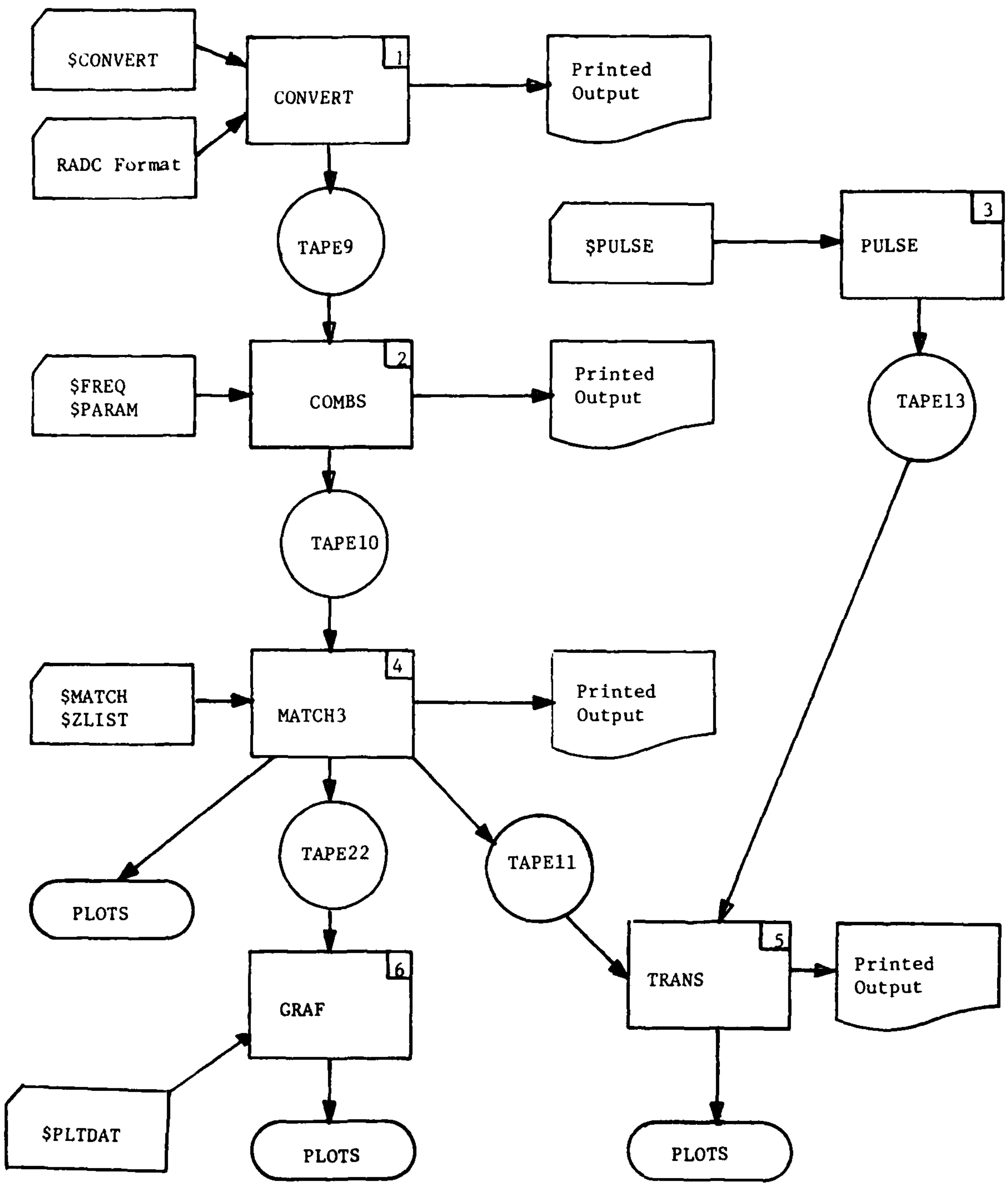
Size of the first matrix.

APPENDIX B

Saw Device Analysis Run Deck and Data Flow Chart

```
ELTC1,CM220000,T40.                                              2582     ELTERMAN
ATTACH,CONVERT,CONVERT2X3716,ID=SLOBOD,MR=1.
ATTACH,COMBS,COMBS3AX3716,ID=DOLAN,MR=1.
ATTACH,MATCH,MATCH3,ID=SLOBOD,MR=1.
ATTACH,GRAF,GRAF,ID=SLOBOD,MR=1.
ATTACH,TRANS,TRANSX3716,ID=SLOBOD,MR=1.
ATTACH,PULSE,PULSEX3716,ID=FENSTER,MR=1.
ATTACH,PEN,OFFLINEPEN.
LIBRARY,PEN.
REQUEST,TAPE39,*Q.
DISPOSE,TAPE39,*LR.
CONVERT.
REQUEST,TAPE10,*PF.
LDSET,PRESET=ZERO.
COMBS.
COMMENT.---CATALOG,TAPE10 CARD GOES HERE---
LDSET,PRESET=ZERO.
PULSE.
LDSET,PRESET=ZERO.
MATCH.
LDSET,PRESET=ZERO.
TRANS.
RFL,60000.
LDSET,PRESET=ZERO.
GRAF.
REWIND,TAPE9.
RFL,50000.
COPYSB6,TAPE9.
789789789789789789789789789789789789789789789789789789789789789789789789789789
THIS IS THE TITLE CARD PRECEDING $CONVERT.
 $CONVERT OVALAP=5.08E-3,PADS=3.81E-5,W1=1.27E-5,SINGLE=T,T,DUMMY=F,F $
H0(N) DECK FOR AN UNAPPODIZED COMB WITH 10 GAPS
    -4    1.0000000E00
    -3    1.0000000E00
    -2    1.0000000E00
    -1    1.0000000E00
     0    1.0000000E00
     1    1.0000000E00
     2    1.0000000E00
     3    1.0000000E00
     4    1.0000000E00
     5    1.0000000E00
789789789789789789789789789789789789789789789789789789789789789789789789789789
789789789789789789789789789789789789789789789789789789789789789789789789789789
H0(N) DECK FOR AN UNAPPODIZED COMB WITH 10 GAPS
    -4    1.0000000E00
    -3    1.0000000E00
    -2    1.0000000E00
    -1    1.0000000E00
     0    1.0000000E00
     1    1.0000000E00
     2    1.0000000E00
     3    1.0000000E00
     4    1.0000000E00
     5    1.0000000E00
789789789789789789789789789789789789789789789789789789789789789789789789789789
789789789789789789789789789789789789789789789789789789789789789789789789789789
THIS IS THE COMBS TITLE CARD
 $FREQ FINIT=41.E6,FINC=.25E6,IFREQ=72,IPRT=2,ICOPY=77$
 $PARAM OVERL=5.1562E-3,VEL=3444.,CAP0=2.5E-10,RHO=2392.5$
```

```
     1.015(2E15.8,2(I2,2E15.8),2(I1,2F1.0))
    78978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978
     $PULSES FINIT=41.E6,FINC=.25E6,IFREQ=72,FZERO=50.E6,PWIDTH=1.E-6$
    78978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978
    ELTERMAN 4904 CODE 1 TEST
     $MATCH GRAPH=T,T,T, CHART=T, SCOPE=F,JUNIT=11 $
     $ZLIST L(2)=.7E-6, R(2)=10., IREP=2 $
    78978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978
     $TRANS IUNIT=13,11, SCOPE=F, GRAPH=T $
    78978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978978
     $PLTDAT FZERO=50.E6,DIV=1.E6$
    67896789678967896789678967896789678967896789678967896789678967896789678967896789678967896789678967896789678967896789678967896789
```

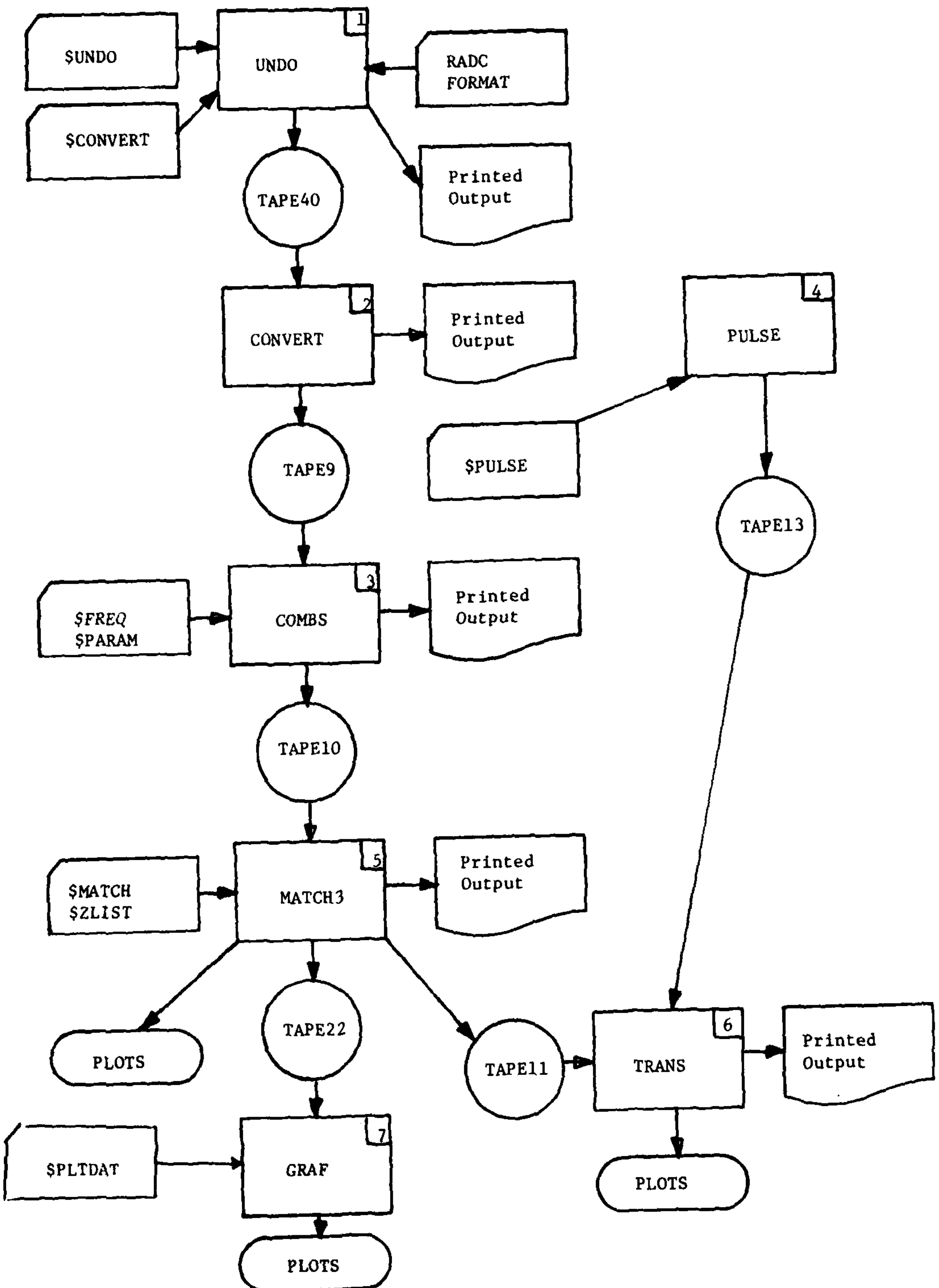

$CONVERT
RADC Format
CONVERT
1
Printed Output
TAPE9
$PULSE
PULSE
3
$FREQ
$PARAM
COMBS
2
Printed Output
TAPE13
TAPE10
$MATCH
$ZLIST
MATCH3
4
Printed Output
TAPE22
TAPE11
PLOTS
GRAF
6
TRANS
5
Printed Output
$PLTDAT
PLOTS
PLOTS

APPENDIX C

Saw Device Analysis Run (With Program UNDO) Deck and Data Flow Chart

```
ELTC1,CM220000,T40.                                        2562     ELTERMAN
ATTACH,UNDO,UNDOX4904,ID=ELTERMAN,MR=1.
ATTACH,CONVERT,CONVERT2X3716,ID=SLOBOD,MR=1.
ATTACH,COMBS,COMBS3AX4904,ID=ELTERMAN,MR=1.
ATTACH,MATCH,MATCH3,ID=SLOBOD,MR=1.
ATTACH,GRAF,GRAF,ID=SLOBOD,MR=1.
ATTACH,TRANS,TRANSX3716,ID=SLOBOD,MR=1.
ATTACH,PULSE,PULSEX3716,ID=FENSTER,MR=1.
ATTACH,PEN,ONLINEPEN.
LIBRARY,PEN.
REQUEST,PLOT,*Q.
DISPOSE,PLOT,*OL.                                                             .TNE
UNDO.
CONVERT,TAPE40.
REQUEST,TAPE10,*PF.
LDSET,PRESET=ZERO.
COMBS.
COMMENT.---CATALOG,TAPE10 CARD GOES HERE---
LDSET,PRESET=ZERO.
PULSE.
LDSET,PRESET=ZERO.
MATCH.
LDSET,PRESET=ZERO.
TRANS.
LDSET,PRESET=ZERO.
GRAF.
REWIND,TAPE9,TAPE28,TAPE29,TAPE10,TAPE11,TAPE13.
COPYSBF,TAPE29.
COPYSBF,TAPE9.
COPYSBF,TAPE29.
COPYSBF,TAPE9.
COPYSBF,TAPE28.
COPYSBF,TAPE10.
COPYSBF,TAPE11.
COPYSBF,TAPE13.
789789789789789789789789789789789789789789789789789789789789789789789789789789789
 $UNDO PERIODC=T,T,T,T,T $
THIS IS THE TITLE CARD PRECEDING $CONVERT.
 $CONVERT OVALAP=5.08E-3,PADS=3.81E-5,W1=1.27E-5,SINGLE=T,T,DUMMY=F,F $
HO(N) DECK FOR AN UNAPPODIZED COMB WITH 10 GAPS
    -4    1.0000000E00
    -3    1.0000000E00
    -2    1.0000000E00
    -1    1.0000000E00
     0    1.0000000E00
     1    1.0000000E00
     2    1.0000000E00
     3    1.0000000E00
     4    1.0000000E00
     5    1.0000000E00
789789789789789789789789789789789789789789789789789789789789789789789789789789789
ACT(M) DECK FOR PERIODIC TRANSDUCER WITH 11 FINGERS (USUALLY NOT INCLUDED)
  -4.5        -1.143E-4
  -3.5        -0.889E-4
  -2.5        -0.635E-4
  -1.5        -0.381E-4
  -0.5        -0.127E-4
   0.5         0.127E-4
   1.5         0.381E-4
   2.5         0.635E-4
   3.5         0.889E-4
```

```
4.5         1.145E-4
5.5         1.397E-4
789789789789789789789789789789789789789789789789789789789789789789789789789789
H0(N) DECK FOR AN UNAPPODIZED COMB WITH 10 GAPS
  -4    1.0000000E00
  -3    1.0000000E00
  -2    1.0000000E00
  -1    1.0000000E00
   0    1.0000000E00
   1    1.0000000E00
   2    1.0000000E00
   3    1.0000000E00
   4    1.0000000E00
   5    1.0000000E00
789789789789789789789789789789789789789789789789789789789789789789789789789789
789789789789789789789789789789789789789789789789789789789789789789789789789789
THIS IS THE COMBS TITLE CARD
 $FREQ FINIT=41.E6,FINC=.25E6,IFREQ=72,IPRT=2,ICOPY=77$
 $PARAM OVERL=5.15F2E-3,VEL=3484,CAPO=2.5E-10,RHO=2382.5$
    1.015(2E15.8,2(I2,2F15.8),2(I1,2F1.0))
789789789789789789789789789789789789789789789789789789789789789789789789789789
 $PULSES FINIT=41.E6,FINC=.25E6,IFREQ=72,FZERO=50.E6,PWIDTH=1.E-6$
789789789789789789789789789789789789789789789789789789789789789789789789789789
DOLAN PLOT X4904
 $MATCH GRAPH=T,T,T, CHART=T, SCOPE=F,JUNIT=11 $
 $ZLIST L(2)=.7E-6, R(2)=10., IREP=2 $
789789789789789789789789789789789789789789789789789789789789789789789789789789
 $TRANS IUNIT=13,11, SCOPE=F, GRAPH=T $
789789789789789789789789789789789789789789789789789789789789789789789789789789
 $PLTDAT FZERO=50.E6,DIV=1.E6$
789789789789789789789789789789789789789789789789789789789789789789789789789789
ATTACH,COMBS,COMBS3AX3716,ID=DOLAN,MR=1.
 $UNDO BETA0=50.82,Z70=914.82,FZERO=331.0E6 $
 $UNDO BETA0=50.82,Z70=914.82,FZERO=321.0E6,HASIS=F,T,F,F,F $
678967896789678967896789678967896789678967896789678967896789678967896789678967
```


$UNDO
UNDO
1
RADC FORMAT
$CONVERT
TAPE40
Printed Output
CONVERT
2
Printed Output
PULSE
4
TAPE9
$PULSE
TAPE13
$FREQ $PARAM
COMBS
3
Printed Output
TAPE10
$MATCH $ZLIST
MATCH3
5
Printed Output
PLOTS
TAPE22
TAPE11
TRANS
6
Printed Output
$PLTDAT
GRAF
7
PLOTS
PLOTS

APPENDIX D

Step and Repeat Machine Tape Generation Run (with Programs HUGHES and LETTER and with a physical tape) and Data Flow Chart

```
SLOBB,CM160000,T400,TP1,NT1.                                2586    SLOBODNIK   01
VSN,TAPE39=WTPLOT,TAPE49=28MA81/NT.
PAUSE,PLS MOUNT ON 9 TRACK DRIVE.
REQUEST,TAPE49,NT,HD,S,EB,RING.  (28MA81/SLOBOD)
REQUEST,TAPE39,PING.  (WTPLOT)
PAUSE. PLS TYPE IN WT NUMBER OF PLOT TAPE
ATTACH,CONVERT,CONVERT2X3716,ID=SLOBODNIK,MR=1.
ATTACH,HUGHES,HUGHES2X3716,ID=SLOBODNIK,MR=1.
ATTACH,HUESCHK,HUESCHKX3716,ID=SLOBODNIK,MR=1.
ATTACH,LETTER,LETTERX3716,ID=SLOBODNIK,MR=1.
COMMENT.                    .DOBOLS=DI,SSECCAON=WP,6173X1RFTSET,TOLPRT,H
ATTACH,PEN,OFFLINEPEN.
LIBRARY,PEN.
CONVERT,PL=50000.
COPYBR,INPUT,X.
LETTER,X.
LDSET,PRESET=ZERO.
HUGHES,X,PL=50000.
LDSET,PRESET=ZERO.
HUESCHK.
LDSET,PRESET=ZERO.
COMMENT.
PAUSE.NOTE - PLOT TAPE CONTAINS MORE THAN ONE PLOT
789789789789789789789789789789789789789789789789789789789789789789789789789789
28MA81 WW OF TWOFZT=140 CSP. USE TO GET DELTERV FOR LTO MDC.
 $CONVERT NCOMBS=1, OVALAP=600.0E-6, PADS=5.0E-6, W1=1.276E-6, W2=1.276E-6,
 DUMMY=F,F,F $
HO(N) DECK FOR WW DERIVED FROM 140 CSP   DELTERV=0.0, W1=W2=1.276E-6
  -87  .10000000E+01
  -86  .10000000E+01
  -85 0.
  -84 -.10000000E+01
  -83 0.
  -82 0.
  -81 -.10000000E+01
  -80 0.
  -79 0.
  -78 -.10000000E+01
  -77 0.
  -76 0.
  -75 -.10000000E+01
  -74 0.
  -73 0.
  -72 -.10000000E+01
  -71 0.
  -70 0.
  -69 -.10000000E+01
  -68 -.10000000E+01
  -67 0.
  -66 0.
  -65 -.10000000E+01
  -64 -.10000000E+01
  -63 -.10000000E+01
  -62 0.
  -61 0.
  -60 -.10000000E+01
  -59 -.10000000E+01
  -58 -.10000000E+01
  -57 -.10000000E+01
  -56 -.10000000E+01
  -55 -.10000000E+01
```

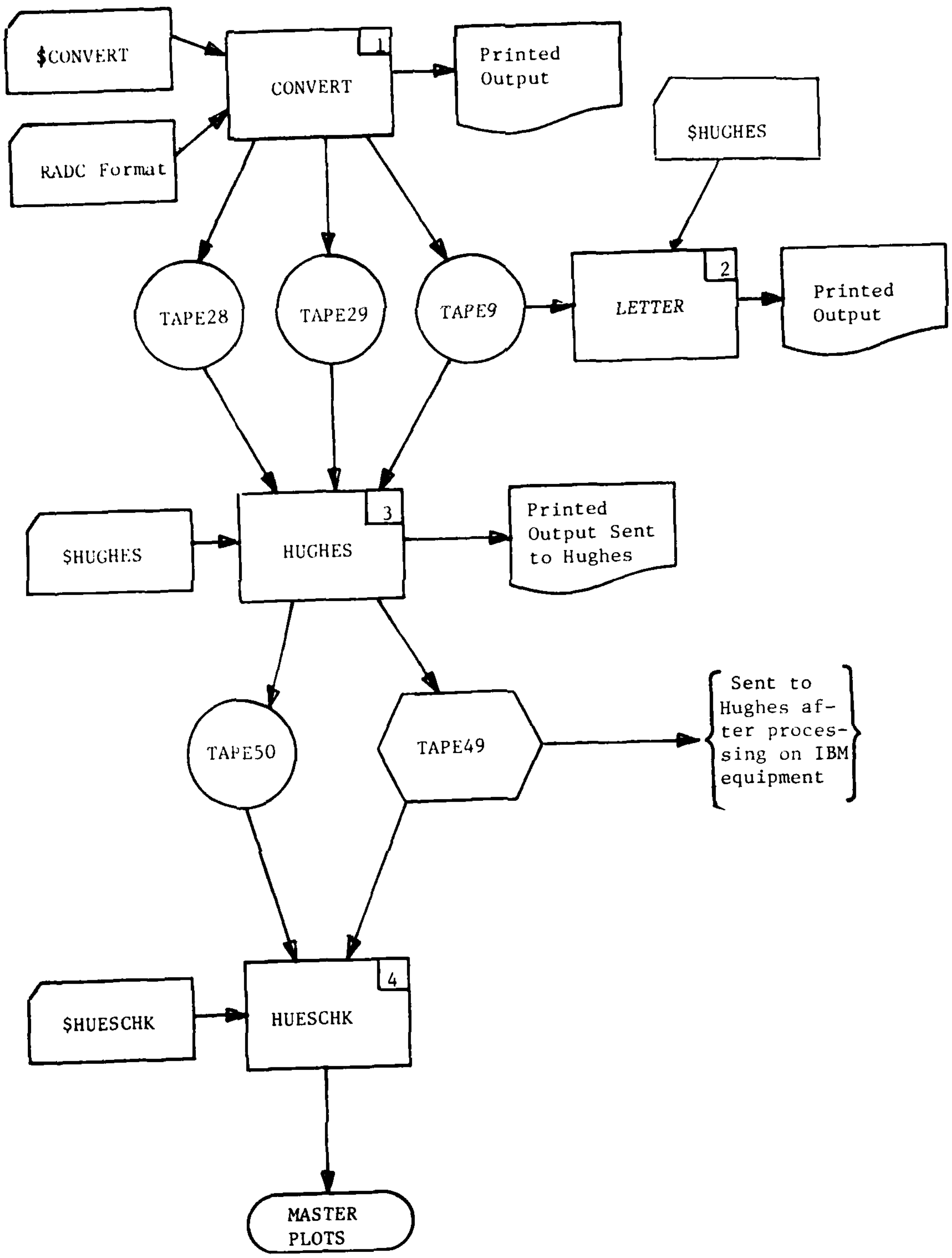
-54	-.10000000E+01
-53	-.10000000E+01
-52	-.10000000E+01
-51	-.10000000E+01
-50	-.10000000E+01
-49	-.10000000E+01
-48	-.10000000E+01
-47	-.10000000E+01
-46	-.10000000E+01
-45	-.10000000E+01
-44	-.10000000E+01
-43	-.10000000E+01
-42	-.10000000E+01
-41	-.10000000E+01
-40	-.10000000E+01
-39	-.10000000E+01
-38	-.10000000E+01
-37	-.10000000E+01
-36	-.10000000E+01
-35	-.10000000E+01
-34	-.10000000E+01
-33	-.10000000E+01
-32	-.10000000E+01
-31	-.10000000E+01
-30	-.10000000E+01
-29	-.10000000E+01
-28	-.10000000E+01
-27	-.10000000E+01
-26	0.
-25	.10000000E+01
-24	.10000000E+01
-23	.10000000E+01
-22	.10000000E+01
-21	.10000000E+01
-20	.10000000E+01
-19	.10000000E+01
-18	.10000000E+01
-17	.10000000E+01
-16	.10000000E+01
-15	.10000000E+01
-14	.10000000E+01
-13	.10000000E+01
-12	.10000000E+01
-11	.10000000E+01
-10	.10000000E+01
-9	.10000000E+01
-8	.10000000E+01
-7	.10000000E+01
-6	.10000000E+01
-5	.10000000E+01
-4	.10000000E+01
-3	.10000000E+01
-2	.10000000E+01
-1	.10000000E+01
0	.10000000E+01
1	.10000000E+01
2	.10000000E+01
3	.10000000E+01
4	.10000000E+01
5	.10000000E+01
6	.10000000E+01
7	.10000000E+01
8	.10000000E+01
9	.10000000E+01
10	.10000000E+01
11	.10000000E+01

12	.10000000E+01
13	.10000000E+01
14	.10000000E+01
15	.10000000E+01
16	.10000000E+01
17	.10000000E+01
18	.10000000E+01
19	.10000000E+01
20	.10000000E+01
21	.10000000E+01
22	.10000000E+01
23	.10000000E+01
24	.10000000E+01
25	.10000000E+01
26	.10000000E+01
27	0.
28	-.10000000E+01
29	-.10000000E+01
30	-.10000000E+01
31	-.10000000E+01
32	-.10000000E+01
33	-.10000000E+01
34	-.10000000E+01
35	-.10000000E+01
36	-.10000000E+01
37	-.10000000E+01
38	-.10000000E+01
39	-.10000000E+01
40	-.10000000E+01
41	-.10000000E+01
42	-.10000000E+01
43	-.10000000E+01
44	-.10000000E+01
45	-.10000000E+01
46	-.10000000E+01
47	-.10000000E+01
48	-.10000000E+01
49	-.10000000E+01
50	-.10000000E+01
51	-.10000000E+01
52	-.10000000E+01
53	-.10000000E+01
54	-.10000000E+01
55	-.10000000E+01
56	-.10000000E+01
57	-.10000000E+01
58	-.10000000E+01
59	-.10000000E+01
60	-.10000000E+01
61	-.10000000E+01
62	0.
63	0.
64	-.10000000E+01
65	-.10000000E+01
66	-.10000000E+01
67	0.
68	0.
69	-.10000000E+01
70	-.10000000E+01
71	0.
72	0.
73	-.10000000E+01
74	0.
75	0.
76	-.10000000E+01
77	0.

```
      78 0.
      79 -.10000000E+01
      80 0.
      81 0.
      82 -.10000000E+01
      83 0.
      84 0.
      85 -.10000000E+01
      86 0.
      87  .10000000E+01
      88  .10000000E+01
789789789789789789789789789789789789789789789789789789789789789789789789789
ACT (M) DECK FOR WW DERIVED FROM140 CSP   DELTERV=0.0, W1=W2=1.276E-6
   -87.5 -.67204440E-03
   -86.5 -.66694040E-03
   -85.5 -.60035164E-03
   -84.5 -.59012390E-03
   -83.5 -.58501990E-03
   -82.5 -.55429725E-03
   -81.5 -.54406952E-03
   -80.5 -.53896551E-03
   -79.5 -.50824286E-03
   -78.5 -.49801513E-03
   -77.5 -.49291112E-03
   -76.5 -.48268339E-03
   -75.5 -.47245566E-03
   -74.5 -.46735165E-03
   -73.5 -.45712392E-03
   -72.5 -.44689619E-03
   -71.5 -.44179219E-03
   -70.5 -.43156445E-03
   -69.5 -.42133672E-03
   -68.5 -.41623272E-03
   -67.5 -.41112871E-03
   -66.5 -.40090098E-03
   -65.5 -.39067325E-03
   -64.5 -.38556924E-03
   -63.5 -.38046524E-03
   -62.5 -.37536124E-03
   -61.5 -.36513350E-03
   -60.5 -.35490577E-03
   -59.5 -.34980177E-03
   -58.5 -.34469776E-03
   -57.5 -.32934630E-03
   -56.5 -.32424230E-03
   -55.5 -.31913829E-03
   -54.5 -.31403429E-03
   -53.5 -.29868283E-03
   -52.5 -.29357882E-03
   -51.5 -.28847482E-03
   -50.5 -.28337082E-03
   -49.5 -.27826681E-03
   -48.5 -.27316281E-03
   -47.5 -.25781135E-03
   -46.5 -.25270734E-03
   -45.5 -.24760334E-03
   -44.5 -.24249934E-03
   -43.5 -.23739533E-03
   -42.5 -.23229133E-03
   -41.5 -.22718733E-03
   -40.5 -.22208332E-03
   -39.5 -.21697932E-03
   -38.5 -.20162786E-03
   -37.5 -.19652386E-03
   -36.5 -.19141985E-03
   -35.5 -.18631585E-03
```

-34.5	-.18121185E-03
-33.5	-.17610784E-03
-32.5	-.17100384E-03
-31.5	-.16589984E-03
-30.5	-.16079583E-03
-29.5	-.15569183E-03
-28.5	-.15058783E-03
-27.5	-.14548382E-03
-26.5	-.14037982E-03
-25.5	-.13015209E-03
-24.5	-.12504808E-03
-23.5	-.11994408E-03
-22.5	-.11484008E-03
-21.5	-.10973607E-03
-20.5	-.10463207E-03
-19.5	-.99528065E-04
-18.5	-.94424062E-04
-17.5	-.89320059E-04
-16.5	-.84216055E-04
-15.5	-.79112052E-04
-14.5	-.74008049E-04
-13.5	-.68904045E-04
-12.5	-.63800042E-04
-11.5	-.58696039E-04
-10.5	-.53592035E-04
-9.5	-.48488032E-04
-8.5	-.43384028E-04
-7.5	-.38280025E-04
-6.5	-.33176022E-04
-5.5	-.28072018E-04
-4.5	-.22968015E-04
-3.5	-.17864012E-04
-2.5	-.12760008E-04
-1.5	-.76560050E-05
-.5	-.25520017E-05
.5	.25520017E-05
1.5	.76560050E-05
2.5	.12760008E-04
3.5	.17864012E-04
4.5	.22968015E-04
5.5	.28072018E-04
6.5	.33176022E-04
7.5	.38280025E-04
8.5	.43384028E-04
9.5	.48488032E-04
10.5	.53592035E-04
11.5	.58696039E-04
12.5	.63800042E-04
13.5	.68904045E-04
14.5	.74008049E-04
15.5	.79112052E-04
16.5	.84216055E-04
17.5	.89320059E-04
18.5	.94424062E-04
19.5	.99528065E-04
20.5	.10463207E-03
21.5	.10973607E-03
22.5	.11484008E-03
23.5	.11994408E-03
24.5	.12504808E-03
25.5	.13015209E-03
26.5	.13525609E-03
27.5	.14548382E-03
28.5	.15058783E-03
29.5	.15569183E-03
30.5	.16079583E-03

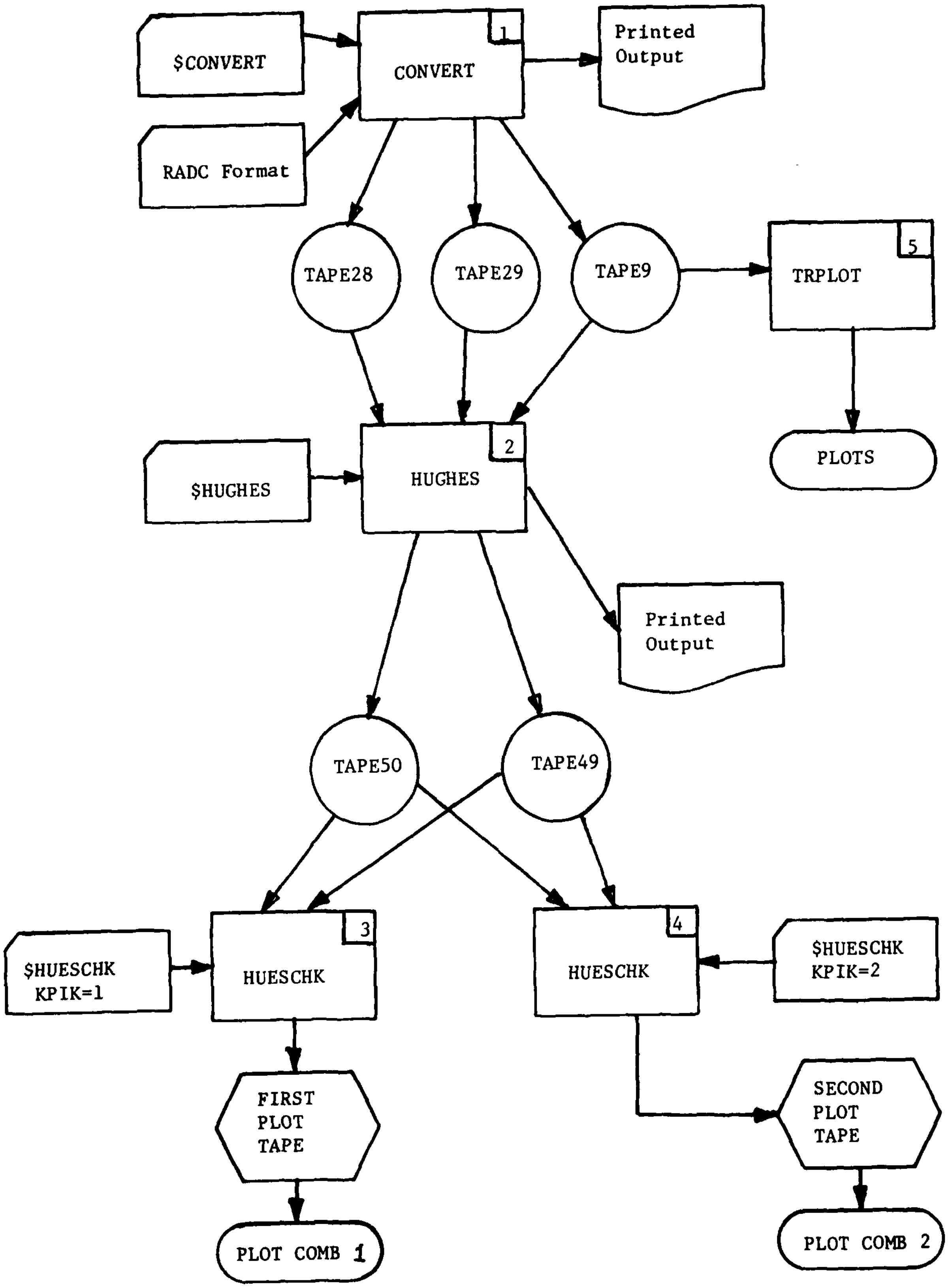
```
31.5   .16589984E-03
32.5   .17100384E-03
33.5   .17610784E-03
34.5   .18121185E-03
35.5   .18631585E-03
36.5   .19141985E-03
37.5   .19652386E-03
38.5   .20162786E-03
39.5   .20673186E-03
40.5   .22208332E-03
41.5   .22718733E-03
42.5   .23229133E-03
43.5   .23739533E-03
44.5   .24249934E-03
45.5   .24760334E-03
46.5   .25270734E-03
47.5   .25781135E-03
48.5   .26291535E-03
49.5   .27826681E-03
50.5   .28337082E-03
51.5   .28847482E-03
52.5   .29357882E-03
53.5   .29868283E-03
54.5   .30378683E-03
55.5   .31913829E-03
56.5   .32424230E-03
57.5   .32934630E-03
58.5   .33445030E-03
59.5   .34980177E-03
60.5   .35490577E-03
61.5   .36000977E-03
62.5   .37023751E-03
63.5   .38046524E-03
64.5   .38556924E-03
65.5   .39067325E-03
66.5   .39577725E-03
67.5   .40600498E-03
68.5   .41623272E-03
69.5   .42133672E-03
70.5   .42644072E-03
71.5   .43666846E-03
72.5   .44689619E-03
73.5   .45200019E-03
74.5   .46222793E-03
75.5   .47245566E-03
76.5   .47755966E-03
77.5   .48778739E-03
78.5   .49801513E-03
79.5   .50311913E-03
80.5   .51334686E-03
81.5   .54406952E-03
82.5   .54917352E-03
83.5   .55940125E-03
84.5   .59012390E-03
85.5   .59522791E-03
86.5   .60545564E-03
87.5   .67204440E-03
88.5   .67714840E-03
789789789789789789789789789789789789789789789789789789789789789789789789789
28MAR81 WW OF TWOFZT=140 CSP. USE TO GET DELTERV FOR LTO MDC.
 $HUGHES RET=6.1E-3, KEARNS=F $
678967896789678967896789678967896789678967896789678967896789678967896789678
```


$CONVERT
RADC Format
CONVERT
1
Printed Output
$HUGHES
TAPE28
TAPE29
TAPE9
LETTER
2
Printed Output
$HUGHES
HUGHES
3
Printed Output Sent to Hughes
TAPE50
TAPE49
Sent to Hughes after processing on IBM equipment
$HUESCHK
HUESCHK
4
MASTER PLOTS

APPENDIX E

Step and Repeat Machine Tape Generation Run (using no physical TAPE49 and the separate plot tape option) and Data Flow Chart

```
ELTER,CM160000,T35,TP1.                              2582     ELTERMAN
ATTACH,TRPLOT,TRPLOTX3716,ID=SLOBOD,MR=1.
ATTACH,HUGHES,HUGHES2X3716,ID=SLOBOD,MR=1.
ATTACH,CONVERT2X3716,ID=SLOBOD,MR=1.
ATTACH,HUESCHK,HBIN,ID=ELTERMAN,MR=1.
ATTACH,PEN,OFFLINEPEN.
LIBRARY,PEN.
VSN,TAPE39=WTPLOT.
REQUEST,TAPE39,RING.
PAUSE.  PLS TYPE IN WT NUMBER OF PLOT TAPE.
LDSET,PRESET=ZERO.
CONVERT.
LDSET,PRESET=ZERO.
HUGHES.
LDSET,PRESET=ZERO.
HUESCHK.
REWIND,TAPE49.
REWIND,TAPE50.
UNLOAD,TAPE39.
PAUSE. PLS MOUNT A DIFFERENT WTPLOT WORK TAPE, TYPE IN WT NUMBER. THANKS.
VSN,TAPE39=WTPLOT.
REQUEST,TAPE39,RING.
LDSET,PRESET=ZERO.
HUESCHK.
UNLOAD,TAPE39.
PAUSE. PLS MOUNT A THIRD WTPLOT WORK TAPE, TYPE IN WT NUMBER. THANKS LOT
VSN,TAPE39=WTPLOT.
REQUEST,TAPE39,RING.
LDSET,PRESET=ZERO.
TRPLOT.
78978978978978978978978978978978978978978978978978978978978978978978978978978978
 $CONVERT OVALAP=10.16E-3,PADS=1.016E-3,W1=.254E-3,
 W2=.254E-3,W3=1.016E-3,SINGLE=T,T,DUMMY=F,F $
 FOUR FINGER UNAPODIZED XDUCER.
   -1    1.0000000E00
    0    1.0000000E00
    1    1.0000000E00
78978978978978978978978978978978978978978978978978978978978978978978978978978978
78978978978978978978978978978978978978978978978978978978978978978978978978978978
SIX FINGER UNAPADIZED TRANSDUCER
   -1    1.0000000E00
    0    1.0000000E00
    1    1.0000000E00
    2    1.0000000E00
    3    1.0000000E00
78978978978978978978978978978978978978978978978978978978978978978978978978978978
78978978978978978978978978978978978978978978978978978978978978978978978978978978
TEST01
 $HUGHES MASTER=1.,PAD2PAD=4.064E-3,PADWID=5.08E-3,
 RET=11.176E-3, KEARNS=T $
78978978978978978978978978978978978978978978978978978978978978978978978978978978
ELTERMAN    K=1      ABC
 $HUESCHK KPIK=1, MAG=2. $
78978978978978978978978978978978978978978978978978978978978978978978978978978978
ELTERMAN     K=2     ABC
 $HUESCHK KPIK=2, MAG=2. $
78978978978978978978978978978978978978978978978978978978978978978978978978978978
ELTERMAN     TRPLOT TEST
67896789678967896789678967896789678967896789678967896789678967896789678967896789
```


$CONVERT
CONVERT
1
Printed Output
RADC Format
TAPE28
TAPE29
TAPE9
TRPLOT
5
PLOTS
$HUGHES
HUGHES
2
Printed Output
TAPE50
TAPE49
$HUESCHK KPIK=1
HUESCHK
3
HUESCHK
4
$HUESCHK KPIK=2
FIRST PLOT TAPE
SECOND PLOT TAPE
PLOT COMB 1
PLOT COMB 2

APPENDIX F

CONVERTXY, HUGHESXY Run (using physical TAPE49) and Data Flow Chart

```
COLVH,CM255000,T60,TP1,NT1.                                2703    COLVIN
REQUEST,TAPE48,*PF.
REQUEST,TAPE50,*PF.
VSN,TAPE39=WTPLOT,TAPE49=25NO71/NT.                              .TN/17ES10=94EPA
PAUSE,PLS MOUNT ON 9 TRACK DRIVE.
REQUEST,TAPE49,NT,HD,S,EB,RING.   (25NO71/COLVIN)
REQUEST,TAPE39,RING.                                                         .TN
PAUSE. PLS TYPE IN WT NUMBER OF PLOT TAPE                                    .TN
ATTACH,CONVERT,CONVXYBIN,ID=ELTERMAN,MR=1.
ATTACH,TRPLOT,TRPLOT,ID=COLVIN,MR=1.
ATTACH,HUGHES,HUGHESXY,ID=COLVIN,MR=1.
ATTACH,HUESCHK,HCHECKBIN,ID=ELTERMAN,MR=1.
ATTACH,PEN,OFFLINEPEN.
LIBRARY,PEN.
COMMENT.                                                          .0*,93EPAT,TS
CONVERT.
LDSET,PRESET=ZERO.
HUGHES.
REWIND,TAPE48.
REWIND,TAPE50.
CATALOG,TAPE48,TAPE49,ID=COLVIN,MR=1.
CATALOG,TAPE50,TAPE50,ID=COLVIN,MR=1.
LDSET,PRESET=ZERO.
HUESCHK.
LDSET,PRESET=ZERO.
TRPLOT.
PAUSE.NOTE - PLOT TAPE CONTAINS TWO (2) PLOTS.
789789789789789789789789789789789789789789789789789789789789789789789789789789
25NO71  449/532 MHZ TRANSDUCERS SIDE BY SIDE
 $CONVERT OVALAP=332.0E-06,332.0E-6,W1=1.02E-6,0.80E-6,DUMMY=F,F,
 SINGLE=T,T,CODE=F,F,PADS=.10E-6,.10E-6,NCOMBS=2 $
27DE71  MULTISTRIP COUPLER   (1/4 WAVE)
   -80  .10000000E+01
   -79  .10000000E+01
   -78  .10000000E+01
   -77  .10000000E+01
   -76  .10000000E+01
   -75  .10000000E+01
   -74  .10000000E+01
   -73  .10000000E+01
   -72  .10000000E+01
   -71  .10000000E+01
   -70  .10000000E+01
   -69  .10000000E+01
   -68  .10000000E+01
   -67  .10000000E+01
   -66  .10000000E+01
   -65  .10000000E+01
   -64  .10000000E+01
   -63  .10000000E+01
   -62  .10000000E+01
   -61  .10000000E+01
   -60  .10000000E+01
   -59  .10000000E+01
   -58  .10000000E+01
   -57  .10000000E+01
   -56  .10000000E+01
   -55  .10000000E+01
   -54  .10000000E+01
   -53  .10000000E+01
   -52  .10000000E+01
```

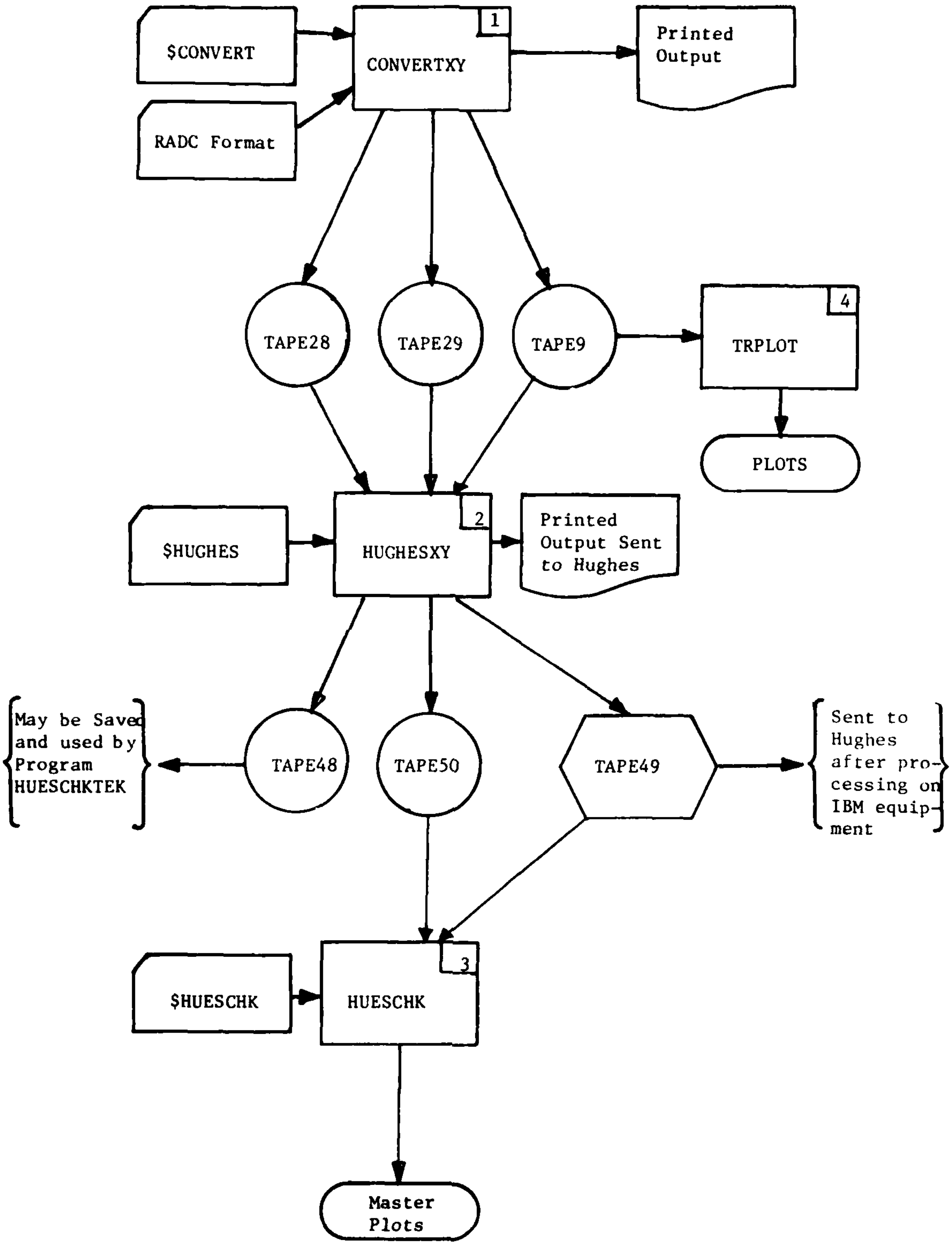
-51	.10000000E+01
-50	.10000000E+01
-49	.10000000E+01
-48	.10000000E+01
-47	.10000000E+01
-46	.10000000E+01
-45	.10000000E+01
-44	.10000000E+01
-43	.10000000E+01
-42	.10000000E+01
-41	.10000000E+01
-40	.10000000E+01
-39	.10000000E+01
-38	.10000000E+01
-37	.10000000E+01
-36	.10000000E+01
-35	.10000000E+01
-34	.10000000E+01
-33	.10000000E+01
-32	.10000000E+01
-31	.10000000E+01
-30	.10000000E+01
-29	.10000000E+01
-28	.10000000E+01
-27	.10000000E+01
-26	.10000000E+01
-25	.10000000E+01
-24	.10000000E+01
-23	.10000000E+01
-22	.10000000E+01
-21	.10000000E+01
-20	.10000000E+01
-19	.10000000E+01
-18	.10000000E+01
-17	.10000000E+01
-16	.10000000E+01
-15	.10000000E+01
-14	.10000000E+01
-13	.10000000E+01
-12	.10000000E+01
-11	.10000000E+01
-10	.10000000E+01
-9	.10000000E+01
-8	.10000000E+01
-7	.10000000E+01
-6	.10000000E+01
-5	.10000000E+01
-4	.10000000E+01
-3	.10000000E+01
-2	.10000000E+01
-1	.10000000E+01
0	.10000000E+01
1	.10000000E+01
2	.10000000E+01
3	.10000000E+01
4	.10000000E+01
5	.10000000E+01
6	.10000000E+01
7	.10000000E+01
8	.10000000E+01
9	.10000000E+01
10	.10000000E+01
11	.10000000E+01
12	.10000000E+01
13	.10000000E+01
14	.10000000E+01

15	.10000000E+01
16	.10000000E+01
17	.10000000E+01
18	.10000000E+01
19	.10000000E+01
20	.10000000E+01
21	.10000000E+01
22	.10000000E+01
23	.10000000E+01
24	.10000000E+01
25	.10000000E+01
26	.10000000E+01
27	.10000000E+01
28	.10000000E+01
29	.10000000E+01
30	.10000000E+01
31	.10000000E+01
32	.10000000E+01
33	.10000000E+01
34	.10000000E+01
35	.10000000E+01
36	.10000000E+01
37	.10000000E+01
38	.10000000E+01
39	.10000000E+01
40	.10000000E+01
41	.10000000E+01
42	.10000000E+01
43	.10000000E+01
44	.10000000E+01
45	.10000000E+01
46	.10000000E+01
47	.10000000E+01
48	.10000000E+01
49	.10000000E+01
50	.10000000E+01
51	.10000000E+01
52	.10000000E+01
53	.10000000E+01
54	.10000000E+01
55	.10000000E+01
56	.10000000E+01
57	.10000000E+01
58	.10000000E+01
59	.10000000E+01
60	.10000000E+01
61	.10000000E+01
62	.10000000E+01
63	.10000000E+01
64	.10000000E+01
65	.10000000E+01
66	.10000000E+01
67	.10000000E+01
68	.10000000E+01
69	.10000000E+01
70	.10000000E+01
71	.10000000E+01
72	.10000000E+01
73	.10000000E+01
74	.10000000E+01
75	.10000000E+01
76	.10000000E+01
77	.10000000E+01
78	.10000000E+01
79	.10000000E+01
80	.10000000E+01

```
789789789789789789789789789789789789789789789789789789789789789789789789789789789789789789789
789789789789789789789789789789789789789789789789789789789789789789789789789789789789789789789
270E71  MULTISTRIP COUPLER   (1/4 WAVE)
   -80  .10000000E+01
   -79  .10000000E+01
   -78  .10000000E+01
   -77  .10000000E+01
   -76  .10000000E+01
   -75  .10000000E+01
   -74  .10000000E+01
   -73  .10000000E+01
   -72  .10000000E+01
   -71  .10000000E+01
   -70  .10000000E+01
   -69  .10000000E+01
   -68  .10000000E+01
   -67  .10000000E+01
   -66  .10000000E+01
   -65  .10000000E+01
   -64  .10000000E+01
   -63  .10000000E+01
   -62  .10000000E+01
   -61  .10000000E+01
   -60  .10000000E+01
   -59  .10000000E+01
   -58  .10000000E+01
   -57  .10000000E+01
   -56  .10000000E+01
   -55  .10000000E+01
   -54  .10000000E+01
   -53  .10000000E+01
   -52  .10000000E+01
   -51  .10000000E+01
   -50  .10000000E+01
   -49  .10000000E+01
   -48  .10000000E+01
   -47  .10000000E+01
   -46  .10000000E+01
   -45  .10000000E+01
   -44  .10000000E+01
   -43  .10000000E+01
   -42  .10000000E+01
   -41  .10000000E+01
   -40  .10000000E+01
   -39  .10000000E+01
   -38  .10000000E+01
   -37  .10000000E+01
   -36  .10000000E+01
   -35  .10000000E+01
   -34  .10000000E+01
   -33  .10000000E+01
   -32  .10000000E+01
   -31  .10000000E+01
   -30  .10000000E+01
   -29  .10000000E+01
   -28  .10000000E+01
   -27  .10000000E+01
   -26  .10000000E+01
   -25  .10000000E+01
   -24  .10000000E+01
   -23  .10000000E+01
   -22  .10000000E+01
   -21  .10000000E+01
   -20  .10000000E+01
   -19  .10000000E+01
   -18  .10000000E+01
```

-17	.10000000E+01
-16	.10000000E+01
-15	.10000000E+01
-14	.10000000E+01
-13	.10000000E+01
-12	.10000000E+01
-11	.10000000E+01
-10	.10000000E+01
-9	.10000000E+01
-8	.10000000E+01
-7	.10000000E+01
-6	.10000000E+01
-5	.10000000E+01
-4	.10000000E+01
-3	.10000000E+01
-2	.10000000E+01
-1	.10000000E+01
0	.10000000E+01
1	.10000000E+01
2	.10000000E+01
3	.10000000E+01
4	.10000000E+01
5	.10000000E+01
6	.10000000E+01
7	.10000000E+01
8	.10000000E+01
9	.10000000E+01
10	.10000000E+01
11	.10000000E+01
12	.10000000E+01
13	.10000000E+01
14	.10000000E+01
15	.10000000E+01
16	.10000000E+01
17	.10000000E+01
18	.10000000E+01
19	.10000000E+01
20	.10000000E+01
21	.10000000E+01
22	.10000000E+01
23	.10000000E+01
24	.10000000E+01
25	.10000000E+01
26	.10000000E+01
27	.10000000E+01
28	.10000000E+01
29	.10000000E+01
30	.10000000E+01
31	.10000000E+01
32	.10000000E+01
33	.10000000E+01
34	.10000000E+01
35	.10000000E+01
36	.10000000E+01
37	.10000000E+01
38	.10000000E+01
39	.10000000E+01
40	.10000000E+01
41	.10000000E+01
42	.10000000E+01
43	.10000000E+01
44	.10000000E+01
45	.10000000E+01
46	.10000000E+01
47	.10000000E+01
48	.10000000E+01

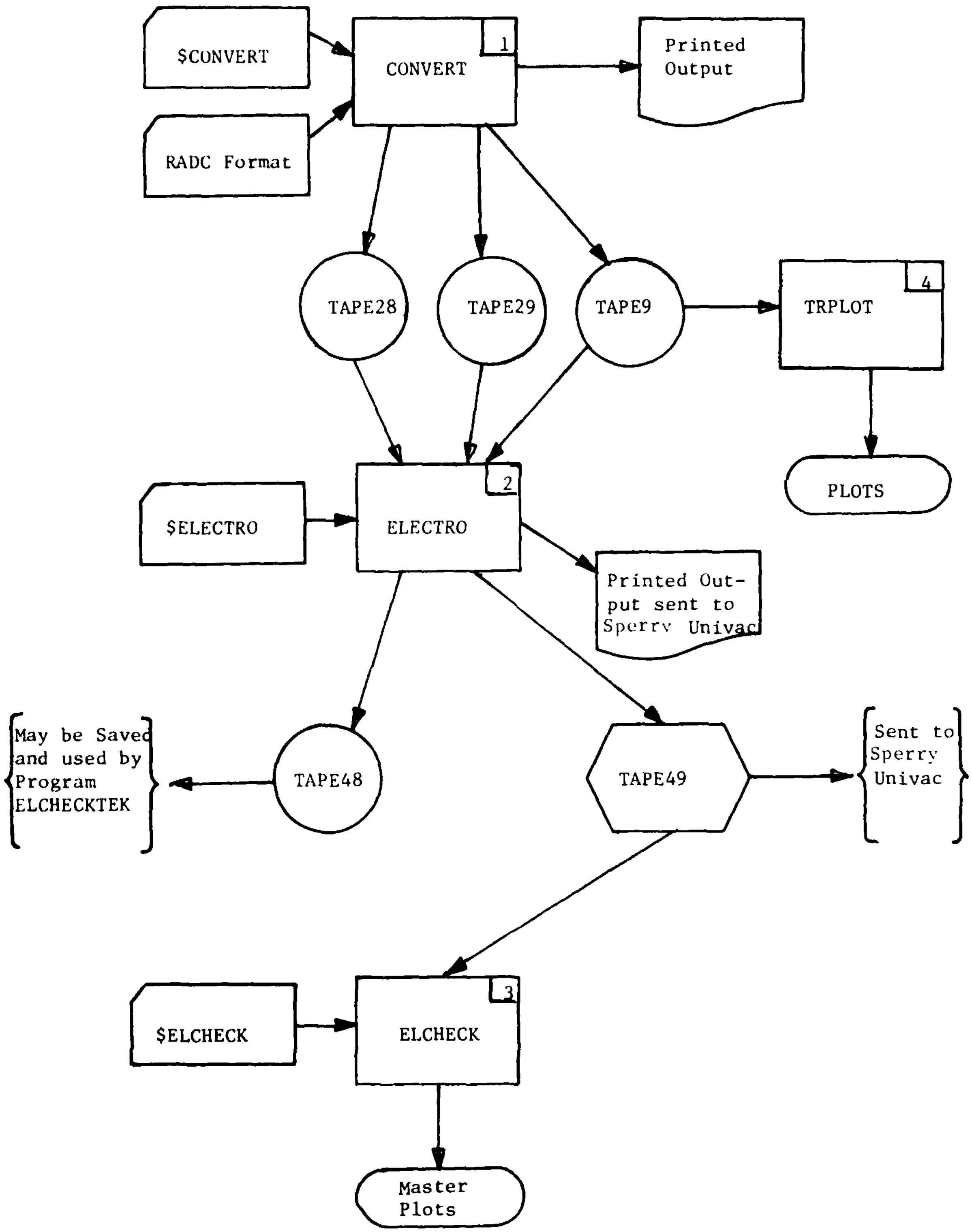
```
49  .10000000E+01
50  .10000000E+01
51  .10000000E+01
52  .10000000E+01
53  .10000000E+01
54  .10000000E+01
55  .10000000E+01
56  .10000000E+01
57  .10000000E+01
58  .10000000E+01
59  .10000000E+01
60  .10000000E+01
61  .10000000E+01
62  .10000000E+01
63  .10000000E+01
64  .10000000E+01
65  .10000000E+01
66  .10000000E+01
67  .10000000E+01
68  .10000000E+01
69  .10000000E+01
70  .10000000E+01
71  .10000000E+01
72  .10000000E+01
73  .10000000E+01
74  .10000000E+01
75  .10000000E+01
76  .10000000E+01
77  .10000000E+01
78  .10000000E+01
79  .10000000E+01
80  .10000000E+01
789789789789789789789789789789789789789789789789789789789789789789789789789
789789789789789789789789789789789789789789789789789789789789789789789789789
25NO71  449/532 MHZ TRANSDUCERS SIDE BY SIDE
 $HUGHES                        NOPAD=T,T,XP=.0254,.0254,
 YP=.029155,.021645 $
789789789789789789789789789789789789789789789789789789789789789789789789789
COLVIN PLOTS X3764
 $HUESCHK HALF=F $
789789789789789789789789789789789789789789789789789789789789789789789789789
25NO71  COLVIN PLOTS X3585 PLOT 2
678967896789678967896789678967896789678967896789678967896789678967896789678
```


$CONVERT
CONVERTXY
1
Printed Output
RADC Format
TAPE28
TAPE29
TAPE9
TRPLOT
4
PLOTS
$HUGHES
HUGHESXY
2
Printed Output Sent to Hughes
May be Saved and used by Program HUESCHKTEK
TAPE48
TAPE50
TAPE49
Sent to Hughes after processing on IBM equipment
$HUESCHK
HUESCHK
3
Master Plots

APPENDIX G

Pattern Generator Tape Generation Run (using physical TAPE49) and Data Flow Chart

```
COLVO,T99,CM132000,TP1,NT1.                                   2703     COLVIN
PURGE,A,TAPE490,ID=COLVIN.
EXIT(U)
VSN,TAPE39=WTPLOT,TAPE49=21FE71/NT.
PAUSE,PLS MOUNT ON 9 TRACK DRIVE.
REQUEST,TAPE49,NT,HD,S,EB,RING.  (21FE71/COLVIN)
REQUEST,TAPE39,RING.
REQUEST,TAPE48,*PF.
PAUSE. PLS TYPE IN WT NUMBER OF PLOT TAPE
ATTACH,CONVERT,CONVERT,ID=COLVIN,MR=1.
ATTACH,ELECTRO,ELECTROC,ID=COLVIN,MR=1.
ATTACH,ELCHECK,ELCHECK,ID=COLVIN,MR=1.
ATTACH,TRPLOT,TRPLOT,ID=COLVIN,MR=1.
ATTACH,PEN,OFFLINEPEN.
LIBRARY,PEN.
CONVERT.
ELECTRO.
REWIND,TAPE48.
CATALOG,TAPE48,TAPE490,ID=COLVIN,MR=1.
LDSET,PRESET=ZERO.
ELCHECK.
LDSET,PRESET=ZERO.
PAUSE.NOTE - PLOT TAPE CONTAINS TWO (2) PLOTS.
TRPLOT.
789789789789789789789789789789789789789789789789789789789789789789789789789789
21FE71  SECOND HARMONIC TRANSDUCER 414 MHZ BANDWIDTH 360-468 MHZ
 $CONVERT OVALAP=131.0E-06,W1=2.08E-06,NCOMBS=1,   DUMMY=F,F,SINGLE=T,T,
 CODE=F,F $
21FE71  SECOND HARMONIC TRANSDUCER 414 MHZ BANDWIDTH 360-468 MHZ
    -2  .10000000E+01
    -1  .10000000E+01
     0  .10000000E+01
     1  .10000000E+01
     2  .10000000E+01
789789789789789789789789789789789789789789789789789789789789789789789789789789
789789789789789789789789789789789789789789789789789789789789789789789789789789
21FE71  SECOND HARMONIC TRANSDUCER 414 MHZ BANDWIDTH 360-468 MHZ
 $ELECTRO METRIC=T,POSITIV=T $
789789789789789789789789789789789789789789789789789789789789789789789789789789
 COLVIN. WT HAS ANOTHER PLOT
 $ELCHECK $
789789789789789789789789789789789789789789789789789789789789789789789789789789
 COLVIN. WT HAS ANOTHER PLOT
678967896789678967896789678967896789678967896789678967896789678967896789678967
```


$CONVERT
RADC Format
CONVERT
1
Printed Output
TAPE28
TAPE29
TAPE9
TRPLOT
4
PLOTS
$ELECTRO
ELECTRO
2
Printed Out-put sent to Sperry Univac
May be Saved and used by Program ELCHECKTEK
TAPE48
TAPE49
Sent to Sperry Univac
$ELCHECK
ELCHECK
3
Master Plots

APPENDIX H

CONVERTXY, ELECTROXY Run (using physical TAPE49) and Data Flow Chart.

```
COLVS,T60,CM255000,TP1,NT1.                                    2703     COLVIN
PURGE,A,TAPE49S,ID=COLVIN.
EXIT(U)
VSN,TAPE39=WTPLOT,TAPE49=280E71/NT.
PAUSE,PLS MOUNT ON 9 TRACK DRIVE.
REQUEST,TAPE49,NT,HD,S,EB,RING.  (280E71/COLVIN)
REQUEST,TAPE39,RING.
REQUEST,TAPE48,*PF.
PAUSE. PLS TYPE IN WT NUMBER OF PLOT TAPE.
ATTACH,CONVERT,CONVXYBIN,ID=ELTERMAN,MR=1.
ATTACH,ELECTRO,ELECTROXYBIN,ID=ELTERMAN,MR=1.
ATTACH,ELCHECK,ELCHECK,ID=COLVIN,MR=1.
ATTACH,TRPLOT,TRPLOT,ID=COLVIN,MR=1.
ATTACH,PEN,OFFLINEPEN.
LIBRARY,PEN.
CONVERT.
ELECTRO.
REWIND,TAPE48.
CATALOG,TAPE48,TAPE48S,ID=COLVIN,MR=1.
LDSET,PRESET=ZERO.
ELCHECK.
LDSET,PRESET=ZERO.
PAUSE,NOTE - PLOT TAPE CONTAINS TWO (2) PLOTS.
TRPLOT.
78978978978978978978978978978978978978978978978978978978978978978978978978978978
ELTERMAN  CONVERT TITLE CARD
 $CONVERT NCOMBS=4,OVALAP=.7E-3,.7E-3,.7E-3,.3E-3,
 PADS=.1E-3,.1E-3,.1E-3,.2E-3,DUMMY=T,F,F,F,SINGLE=F,T,F,T,
 W1=.5E-5,1.42E-6,.2E-5,.2E-5,
 W2=.5E-5,.2E-5,.2E-5,.2E-5, REVERSE=F,F,F,T,
 NMIND=500,500,10,10,NMAXD=50,500,10,10 $
HON TITLE CARD
    -6   .80000000E-01
    -5   .14162831E+00
    -4   .31000000E+00
    -3   .54000000E+00
    -2   .77000000E+00
    -1   .93837169E+00
     0   .10000000E+01
     1   .93837169E+00
     2   .77000000E+00
     3   .54000000E+00
     4   .31000000E+00
     5   .14162831E+00
     6   .80000000E-01
78978978978978978978978978978978978978978978978978978978978978978978978978978978
78978978978978978978978978978978978978978978978978978978978978978978978978978978
78978978978978978978978978978978978978978978978978978978978978978978978978978978
ACT OF M TITLE CARD
   -1.5  -.42630700E-05   .14180000E-05
    -.5  -.14225350E-05   .14225350E-05
     .5   .14270700E-05   .14270700E-05
    1.5   .42857450E-05   .14316050E-05
    2.5   .71534900E-05   .14361400E-05
    3.5   .10030305E-04   .14406750E-05
    4.5   .12916190E-04   .14452100E-05
    5.5   .15811145E-04   .14497450E-05
    6.5   .18715170E-04   .14542800E-05
    7.5   .21628265E-04   .14588150E-05
    8.5   .24550430E-04   .14633500E-05
    9.5   .27481665E-04   .14678850E-05
```

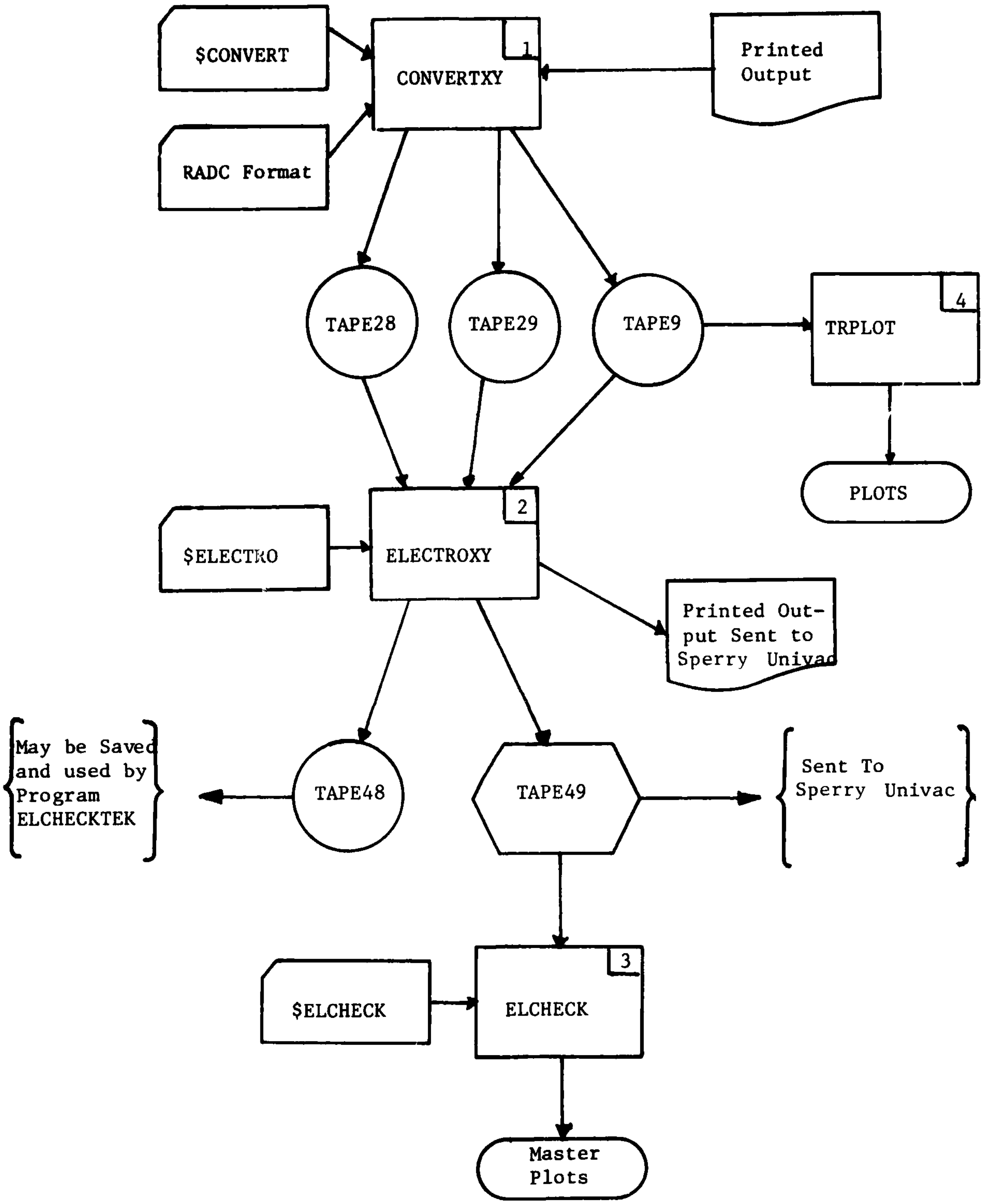
10.5	.30421970E-04	.14724200E-05
11.5	.33371345E-04	.14769550E-05
12.5	.36329790E-04	.14814900E-05
13.5	.39297305E-04	.14860250E-05
14.5	.42273890E-04	.14905600E-05
15.5	.45259545E-04	.14950950E-05
16.5	.48254270E-04	.14996300E-05
17.5	.51258065E-04	.15041650E-05
18.5	.54270930E-04	.15087000E-05
19.5	.57292865E-04	.15132350E-05
20.5	.60323870E-04	.15177700E-05
21.5	.63363945E-04	.15223050E-05
22.5	.66413090E-04	.15268400E-05
23.5	.69471305E-04	.15313750E-05
24.5	.72538590E-04	.15359100E-05
25.5	.75614945E-04	.15404450E-05
26.5	.78700370E-04	.15449800E-05
27.5	.81794865E-04	.15495150E-05
28.5	.84898430E-04	.15540500E-05
29.5	.88011065E-04	.15585850E-05
30.5	.91132770E-04	.15631200E-05
31.5	.94263545E-04	.15676550E-05
32.5	.97403390E-04	.15721900E-05
33.5	.10055231E-03	.15767250E-05
34.5	.10371029E-03	.15812600E-05
35.5	.10687735E-03	.15857950E-05
36.5	.11005347E-03	.15903300E-05
37.5	.11323866E-03	.15948650E-05
38.5	.11643293E-03	.15994000E-05
39.5	.11963626E-03	.16039350E-05
40.5	.12284867E-03	.16084700E-05
41.5	.12607014E-03	.16130050E-05
42.5	.12930069E-03	.16175400E-05
43.5	.13254030E-03	.16220750E-05
44.5	.13578899E-03	.16266100E-05
45.5	.13904674E-03	.16311450E-05
46.5	.14231357E-03	.16356800E-05
47.5	.14558946E-03	.16402150E-05
48.5	.14887443E-03	.16447500E-05
49.5	.15216846E-03	.16492850E-05
50.5	.15547157E-03	.16538200E-05
51.5	.15878374E-03	.16583550E-05
52.5	.16210499E-03	.16628900E-05
53.5	.16543530E-03	.16674250E-05
54.5	.16877469E-03	.16719600E-05
55.5	.17212314E-03	.16764950E-05
56.5	.17548067E-03	.16810300E-05
57.5	.17884726E-03	.16855650E-05
58.5	.18222293E-03	.16901000E-05
59.5	.18560766E-03	.16946350E-05
60.5	.18900147E-03	.16991700E-05
61.5	.19240434E-03	.17037050E-05
62.5	.19581629E-03	.17082400E-05
63.5	.19923730E-03	.17127750E-05
64.5	.20266739E-03	.17173100E-05
65.5	.20610654E-03	.17218450E-05
66.5	.20955477E-03	.17263800E-05
67.5	.21301206E-03	.17309150E-05
68.5	.21647843E-03	.17354500E-05
69.5	.21995386E-03	.17399850E-05
70.5	.22343837E-03	.17445200E-05
71.5	.22693194E-03	.17490550E-05
72.5	.23043459E-03	.17535900E-05
73.5	.23394630E-03	.17581250E-05
74.5	.23746709E-03	.17626600E-05
75.5	.24099694E-03	.17671950E-05

76.5	.24453587E-03	.17717300E-05
77.5	.24808386E-03	.17762650E-05
78.5	.25164093E-03	.17808000E-05
79.5	.25520706E-03	.17853350E-05
80.5	.25878227E-03	.17898700E-05
81.5	.26236654E-03	.17944050E-05
82.5	.26595989E-03	.17989400E-05
83.5	.26956230E-03	.18034750E-05
84.5	.27317379E-03	.18080100E-05
85.5	.27679434E-03	.18125450E-05
86.5	.28042397E-03	.18170800E-05
87.5	.28406266E-03	.18216150E-05
88.5	.28771043E-03	.18261500E-05
89.5	.29136726E-03	.18306850E-05
90.5	.29503317E-03	.18352200E-05
91.5	.29870814E-03	.18397550E-05
92.5	.30239219E-03	.18442900E-05
93.5	.30608530E-03	.18488250E-05
94.5	.30978749E-03	.18533600E-05
95.5	.31349874E-03	.18578950E-05
96.5	.31721907E-03	.18624300E-05
97.5	.32094846E-03	.18669650E-05
98.5	.32468693E-03	.18715000E-05
99.5	.32843446E-03	.18760350E-05
100.5	.33219107E-03	.18805700E-05
101.5	.33595674E-03	.18851050E-05
102.5	.33973149E-03	.18896400E-05
103.5	.34351530E-03	.18941750E-05
104.5	.34730819E-03	.18987100E-05
105.5	.35111014E-03	.19032450E-05
106.5	.35492117E-03	.19077800E-05
107.5	.35874126E-03	.19123150E-05
108.5	.36257043E-03	.19168500E-05
109.5	.36640866E-03	.19213850E-05
110.5	.37025597E-03	.19259200E-05
111.5	.37411234E-03	.19304550E-05
112.5	.37797779E-03	.19349900E-05
113.5	.38185230E-03	.19395250E-05
114.5	.38573589E-03	.19440600E-05
115.5	.38962854E-03	.19485950E-05
116.5	.39353027E-03	.19531300E-05
117.5	.39744106E-03	.19576650E-05
118.5	.40136093E-03	.19622000E-05
119.5	.40528986E-03	.19667350E-05
120.5	.40922787E-03	.19712700E-05
121.5	.41317494E-03	.19758050E-05
122.5	.41713109E-03	.19803400E-05
123.5	.42109630E-03	.19848750E-05
124.5	.42507059E-03	.19894100E-05
125.5	.42905394E-03	.19939450E-05
126.5	.43304637E-03	.19984800E-05
127.5	.43704786E-03	.20030150E-05
128.5	.44105843E-03	.20075500E-05
129.5	.44507806E-03	.20120850E-05
130.5	.44910677E-03	.20166200E-05
131.5	.45314454E-03	.20211550E-05
132.5	.45719139E-03	.20256900E-05
133.5	.46124730E-03	.20302250E-05
134.5	.46531229E-03	.20347600E-05
135.5	.46938634E-03	.20392950E-05
136.5	.47346947E-03	.20438300E-05
137.5	.47756166E-03	.20483650E-05
138.5	.48166293E-03	.20529000E-05
139.5	.48577326E-03	.20574350E-05
140.5	.48989267E-03	.20619700E-05
141.5	.49402114E-03	.20665050E-05

142.5	.49815869E-03	.20710400E-05
143.5	.50230530E-03	.20755750E-05
144.5	.50646099E-03	.20801100E-05
145.5	.51062574E-03	.20846450E-05
146.5	.51479957E-03	.20891800E-05
147.5	.51898246E-03	.20937150E-05
148.5	.52317443E-03	.20982500E-05
149.5	.52737546E-03	.21027850E-05
150.5	.53158557E-03	.21073200E-05
151.5	.53580474E-03	.21118550E-05
152.5	.54003299E-03	.21163900E-05
153.5	.54427030E-03	.21209250E-05
154.5	.54851669E-03	.21254600E-05
155.5	.55277214E-03	.21299950E-05
156.5	.55703667E-03	.21345300E-05
157.5	.56131026E-03	.21390650E-05
158.5	.56559293E-03	.21436000E-05
159.5	.56988456E-03	.21481350E-05
160.5	.57418547E-03	.21526700E-05
161.5	.57849534E-03	.21572050E-05
162.5	.58281429E-03	.21617400E-05
163.5	.58714230E-03	.21662750E-05
164.5	.59147939E-03	.21708100E-05
165.5	.59582554E-03	.21753450E-05
166.5	.60018077E-03	.21798800E-05
167.5	.60454506E-03	.21844150E-05
168.5	.60891843E-03	.21889500E-05
169.5	.61330086E-03	.21934850E-05
170.5	.61769237E-03	.21980200E-05
171.5	.62209294E-03	.22025550E-05
172.5	.62650259E-03	.22070900E-05
173.5	.63092130E-03	.22116250E-05
174.5	.63534909E-03	.22161600E-05
175.5	.63978594E-03	.22206950E-05
176.5	.64423187E-03	.22252300E-05
177.5	.64868686E-03	.22297650E-05
178.5	.65315093E-03	.22343000E-05
179.5	.65762406E-03	.22388350E-05
180.5	.66210627E-03	.22433700E-05
181.5	.66659754E-03	.22479050E-05
182.5	.67109789E-03	.22524400E-05
183.5	.67560730E-03	.22569750E-05
184.5	.68012579E-03	.22615100E-05
185.5	.68465334E-03	.22660450E-05
186.5	.68918997E-03	.22705800E-05
187.5	.69373566E-03	.22751150E-05
188.5	.69829043E-03	.22796500E-05
189.5	.70285426E-03	.22841850E-05
190.5	.70742717E-03	.22887200E-05
191.5	.71200914E-03	.22932550E-05
192.5	.71660019E-03	.22977900E-05
193.5	.72120030E-03	.23023250E-05
194.5	.72580949E-03	.23068600E-05
195.5	.73042774E-03	.23113950E-05
196.5	.73505507E-03	.23159300E-05
197.5	.73969146E-03	.23204650E-05
198.5	.74433693E-03	.23250000E-05

```
7897897897897897897897897897897897897897897897897897897897897897897897897897
7897897897897897897897897897897897897897897897897897897897897897897897897897
7897897897897897897897897897897897897897897897897897897897897897897897897897
7897897897897897897897897897897897897897897897897897897897897897897897897897
7897897897897897897897897897897897897897897897897897897897897897897897897897
66666666 ELTERMAN ELECTRO TITLE CARD
 $ELECTRO MASTER=5.,
 XP=.1E-1,.2E-1,.1E-1,.2E-1,
 YP=.26E-1,.26E-1,.38E-1,.38E-1,
```

```
PADWID=.1E-2,.2E-2,0.0,.2E-2, PADLEN(4)=.1E-4, POSITIV=T $
7897897897897897897897897897897897897897897897897897897897897897897897897897
COLVIN. WT HAS ANOTHER PLOT
$ELCHECK $
7897897897897897897897897897897897897897897897897897897897897897897897897897
COLVIN. WT HAS ANOTHER PLOT
6789678967896789678967896789678967896789678967896789678967896789678967896789
```


$CONVERT
RADC Format
CONVERTXY
1
Printed Output
TAPE28
TAPE29
TAPE9
TRPLOT
4
PLOTS
$ELECTRO
ELECTROXY
2
Printed Out-put Sent to Sperry Univac
May be Saved and used by Program ELCHECKTEK
TAPE48
TAPE49
Sent To Sperry Univac
$ELCHECK
ELCHECK
3
Master Plots

APPENDIX I

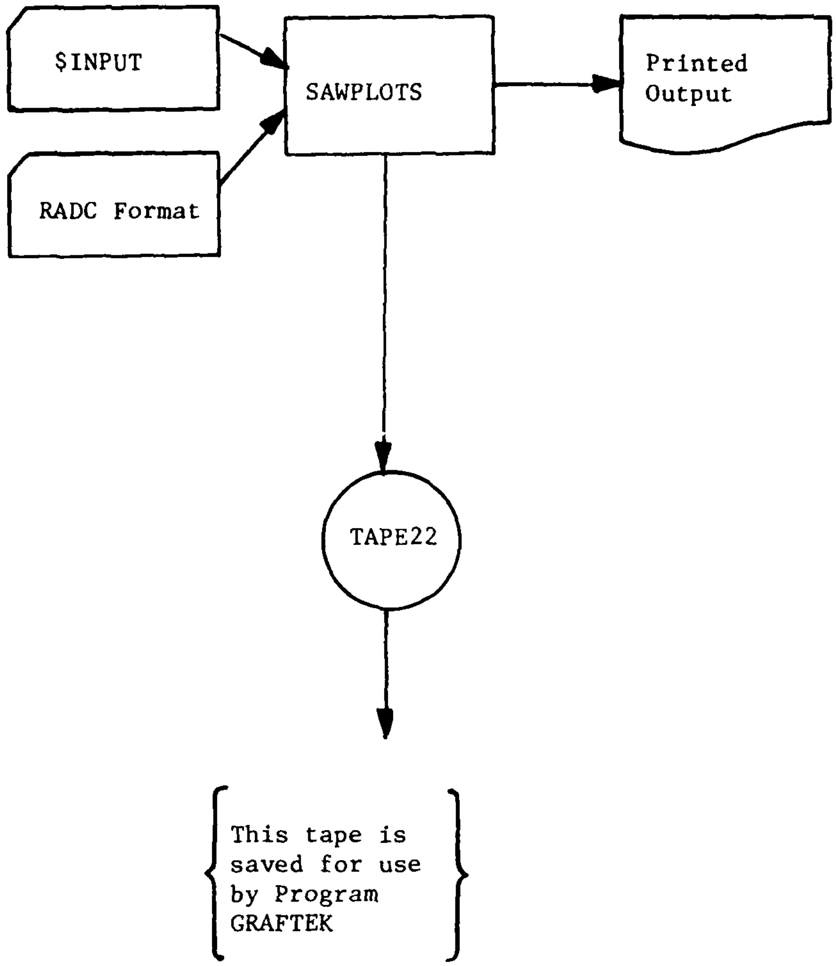
SAWPLOTS Run Deck and Data Flow Chart

```
ELTER,CM65000,T300,STMFA.                              2582    ELTERMAN
PURGE,TA,TAPE22,ID=ELTERMAN.
EXIT(U)
ATTACH,A,SAWB,ID=ELTERMAN,MR=1.
ATTACH,B,DIFF,ID=ELTERMAN,MR=1.
REQUEST,TAPE22,*PF.
LOAD(A)
LOAD(B)
EXECUTE.
CATALOG,TAPE22,TAPE22,ID=ELTERMAN,MR=1.
789789789789789789789789789789789789789789789789789789789789789789789789789789
ELTERMAN, TITLE CARD REF $INPUT
 $INPUT FZERO=340.56,EFSTPR=47.9, DVOVAV=.1033, VS=3230.,GAMMA=-.211,
 ZMETERS=5.48806E-3, DVALAV=2.2154E-4, IN=136,68, OUT=0,0, KOPT=0  $
 PHYSICAL XDUCER  FS-003 2       USE FZERO=344MHZ
  -67  .22518
  -66  .22608
  -65  .22789
  -64  .23060
  -63  .23421
  -62  .23872
  -61  .24368
  -60  .24955
  -59  .25632
  -58  .26354
  -57  .27121
  -56  .27978
  -55  .28926
  -54  .29919
  -53  .30957
  -52  .32040
  -51  .33168
  -50  .34386
  -49  .35650
  -48  .36958
  -47  .38312
  -46  .39711
  -45  .41155
  -44  .42644
  -43  .44179
  -42  .45758
  -41  .47383
  -40  .49007
  -39  .50677
  -38  .52392
  -37  .54106
  -36  .55866
  -35  .57626
  -34  .59386
  -33  .61191
  -32  .62951
  -31  .64711
  -30  .66516
  -29  .68321
  -28  .70081
  -27  .71796
  -26  .73511
  -25  .75226
  -24  .76940
  -23  .78610
  -22  .80190
```

-20	.83258
-19	.84747
-18	.86146
-17	.87500
-16	.88809
-15	.90072
-14	.91291
-13	.92419
-12	.93457
-11	.94404
-10	.95307
-9	.96164
-8	.96931
-7	.97608
-6	.98195
-5	.98691
-4	.99097
-3	.99458
-2	.99729
-1	.99910
0	1.00000
1	1.00000
2	.99910
3	.99729
4	.99458
5	.99097
6	.98691
7	.98195
8	.97608
9	.96931
10	.96164
11	.95307
12	.94404
13	.93457
14	.92419
15	.91291
16	.90072
17	.88809
18	.87500
19	.86146
20	.84747
21	.83258
22	.81724
23	.80190
24	.78610
25	.76940
26	.75226
27	.73511
28	.71796
29	.70081
30	.68321
31	.66518
32	.64711
33	.62951
34	.61191
35	.59386
36	.57626
37	.55866
38	.54106
39	.52392
40	.50677
41	.49007
42	.47383
43	.45758
44	.44179

```
 [illegible]
 46   .41155
 47   .39711
 48   .38312
 49   .36958
 50   .35650
 51   .34386
 52   .33168
 53   .32040
 54   .30957
 55   .29919
 56   .28926
 57   .27978
 58   .27121
 59   .26354
 60   .25632
 61   .24955
 62   .24368
 63   .23872
 64   .23421
 65   .23060
 66   .22789
 67   .22608
 68   .22518
789789789789789789789789789789789789789789789789789789789789789789789789789
789789789789789789789789789789789789789789789789789789789789789789789789789
 UNIFORM XDUCER    FS=0C? 2    NMIN=34,NMAX=33    USE FZERO=344MHZ
-34   .10000000E+01
-33   .10000000E+01
-32   .10000000E+01
-31   .10000000E+01
-30   .10000000E+01
-29   .10000000E+01
-28   .10000000E+01
-27   .10000000E+01
-26   .10000000E+01
-25   .10000000E+01
-24   .10000000E+01
-23   .10000000E+01
-22   .10000000E+01
-21   .10000000E+01
-20   .10000000E+01
-19   .10000000E+01
-18   .10000000E+01
-17   .10000000E+01
-16   .10000000E+01
-15   .10000000E+01
-14   .10000000E+01
-13   .10000000E+01
-12   .10000000E+01
-11   .10000000E+01
-10   .10000000E+01
 -9   .10000000E+01
 -8   .10000000E+01
 -7   .10000000E+01
 -6   .10000000E+01
 -5   .10000000E+01
 -4   .10000000E+01
 -3   .10000000E+01
 -2   .10000000E+01
 -1   .10000000E+01
  0   .10000000E+01
  1   .10000000E+01
  2   .10000000E+01
  3   .10000000E+01
  4   .10000000E+01
```

```
  5  .10000000E+01
  6  .10000000E+01
  7  .10000000E+01
  8  .10000000E+01
  9  .10000000E+01
 10  .10000000E+01
 11  .10000000E+01
 12  .10000000E+01
 13  .10000000E+01
 14  .10000000E+01
 15  .10000000E+01
 16  .10000000E+01
 17  .10000000E+01
 18  .10000000E+01
 19  .10000000E+01
 20  .10000000E+01
 21  .10000000E+01
 22  .10000000E+01
 23  .10000000E+01
 24  .10000000E+01
 25  .10000000E+01
 26  .10000000E+01
 27  .10000000E+01
 28  .10000000E+01
 29  .10000000E+01
 30  .10000000E+01
 31  .10000000E+01
 32  .10000000E+01
 33  .10000000E+01
7897897897897897897897897897897897897897897897897897897897897897897897897897897897
7897897897897897897897897897897897897897897897897897897897897897897897897897897897
ELTERMAN $FREQ TITLE CARD
 $FREQ FINIT=294.E6, FINC=.1E6, IFREQ=1005 $
7897897897897897897897897897897897897897897897897897897897897897897897897897897897
 $PLTDAT FZERO=168.E7, DIV=336.E6 $
7897897897897897897897897897897897897897897897897897897897897897897897897897897897
ATTACH,PEN,OFFLINEPEN.
LIBRARY,PEN.
REQUEST,TAPE39,*Q.
DISPOSE,TAPE39,*LL.
GR.
ATTACH,SIM,SIMPLOTS.
LIBRARY,SIM,IGS274.
ATTACH,GR,GRAF,ID=SLOBOC,MR=1.
6789678967896789678967896789678967896789678967896789678967896789678967896789
```


$INPUT
RADC Format
SAWPLOTS
Printed Output
TAPE22
This tape is saved for use by Program GRAFTEK

APPENDIX J

RATIO Run Deck and Data Flow Chart

AD-A079 623 ANALYSIS AND COMPUTER SYSTEMS INC BURLINGTON MA F/6 17/1
DEVELOPMENT OF NUMERICAL TECHNIQUES AND COMPUTER SYSTEMS FOR DE--ETC(U)
AUG 79 L J ELTERMAN F19628-76-C-0257

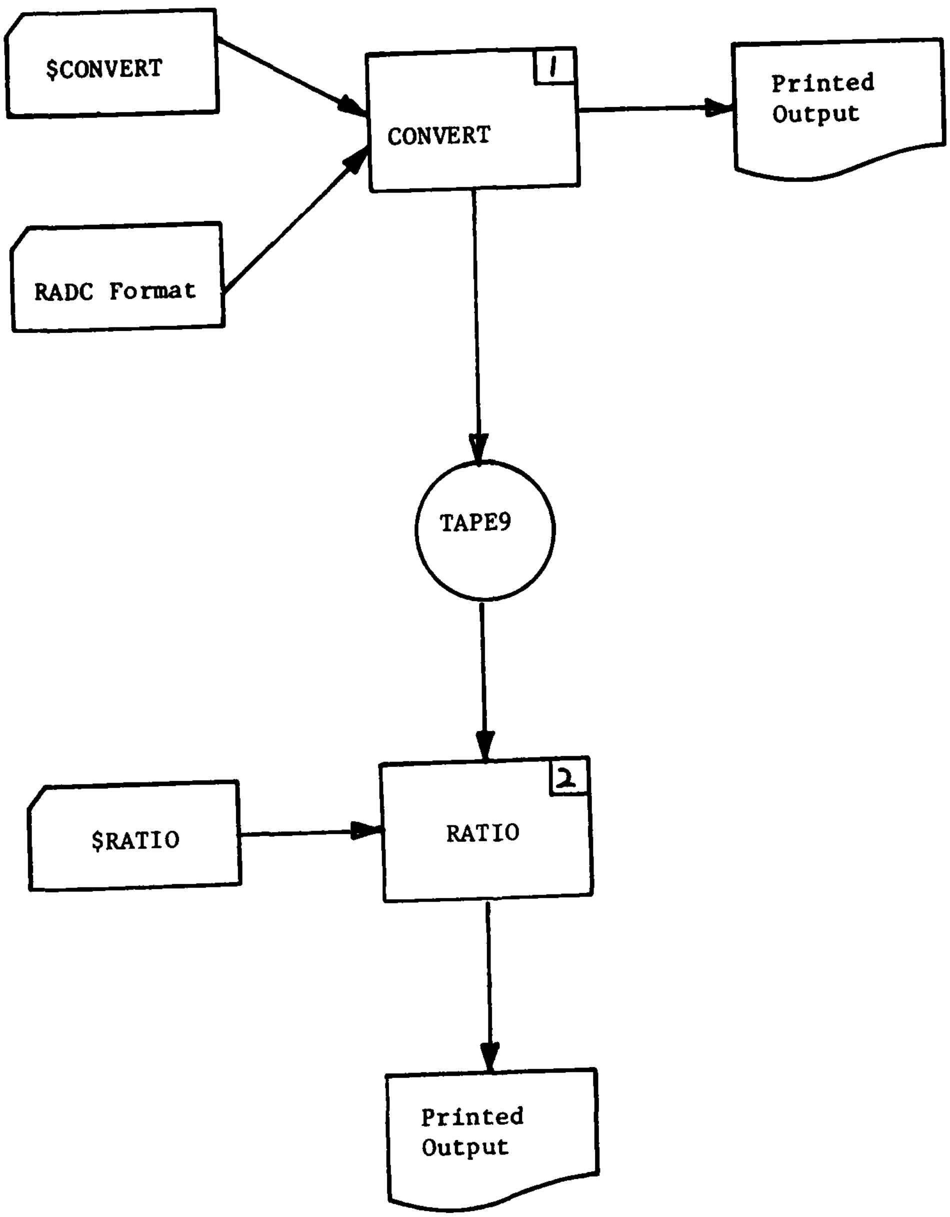
UNCLASSIFIED RADC-TR-79-181 NL

4 OF 4
ADA
079623

END
DATE
FILMED
2-80
DDC

J2

```
ELTER,CM75000,T15.                                            2582     ELTERMAN
ATTACH,CONVERT,CONVERT,ID=COLVIN,MR=1.
ATTACH,RATIO,RATIOBIN,ID=ELTERMAN,MR=1.
CONVERT.
RATIO.
7897897897897897897897897897897897897897897897897897897897897897897897897897897
26AU71    SSB MIXER TRANSDUCER  828 MHZ  BANDWIDTH 720-936 MHZ
 $CONVERT DUMMY=T,F,SINGLE=F,T $
26AU71    SSB MIXER TRANSDUCER  828 MHZ  BANDWIDTH 720-936 MHZ
    -3  .10000000E+01
    -2  .10000000E+01
    -1  .10000000E+01
     0  .10000000E+01
     1  .10000000E+01
     2  .10000000E+01
     3  .10000000E+01
7897897897897897897897897897897897897897897897897897897897897897897897897897897
7897897897897897897897897897897897897897897897897897897897897897897897897897897
26AU71    SSB MIXER TRANSDUCER  828 MHZ  BANDWIDTH 720-936 MHZ
    -3  .10000000E+01
    -2  .10000000E+01
    -1  .10000000E+01
     0  .10000000E+01
     1  .10000000E+01
     2  .10000000E+01
     3  .10000000E+01
7897897897897897897897897897897897897897897897897897897897897897897897897897897
7897897897897897897897897897897897897897897897897897897897897897897897897897897
      THIS IS A REAL LIVE TO TITLE CARD
 $RATIO$
6789678967896789678967896789678967896789678967896789678967896789678967896789
```



☆ U. S. GOVERNMENT PRINTING OFFICE: 1979--600-002--5

APPENDIX K

Program Listings Dealing with Acoustic Prism Studies

```
1                    PRCGRAM TABLE(INPUT,CUTPUT)
                     DIMENSION BETAD(201),PHID(201),PLTID(3)
                     DATA PLTID/30HSZABO  X3683  PROJECT TILT
                     NAMELIST /DETOUR/PHI,GAMMA,DVOVAV,EPS,ITMAX,IER,BETA
5
               C
               C
               C     PRCGRAM TILT
               C     THIS PRCGRAM COMPUTES THE PRISM ANGLE
10             C     REQUIRED FOR A METAL PRISM TC TILT A SURFACE ACOUSTIC
               C     WAVE AS A FUNCTION CF A GIVEN ANGLE
               C     INCLUDING THE EFFECT CF ANISCTROPY
               C
               C
15                   GAMMA=.378
                     EPS=1.E-6
                     ITMAX=100
                     DVOVAV=.00538
                     NPTS=201
20                   EL=.5
                     REAC DETOUR
                     PRINT DETOUR

                     PRINT 99,GAMMA,DVOVAV,EL
25                99 FORMAT(*1GAMMA=*,F6.4,5X*DVOVAV=*F8.7,5X*L=*F5.2,
                   $     //*  PHI(DEG)*4X*BETA(DEG)*6X*W(MM)*/)
                     DO 100 K=1,NPTS
                     PHID(K)=(K-1)/10.
                     PHI=PHID(K)*.01745329252
30                   CALL TILT(PHI,GAMMA,DVOVAV,EPS,ITMAX,IER,BETA)
                     BETAD(K)=BETA*57.29577951
                     W=2.*EL/TAN(BETA)
                 100 PRINT 101,PHID(K),BETAD(K),W
                 101 FCRMAT(F9.3,5X,F8.3,5X,F8.3)
35                   CALL PLTID3(PLTID,100.,12.,1.)
                     CALL DOPLOT(PLTID,PHID,BETAD,NPTS,9HPHI (DEG),9,10HBETA (DEG)
                    $,0)
                     END
```

CARD NR.	SEVERITY	DETAILS	DIAGNOSIS OF FROBLEM
3	I	SZABO X	HOLLERITH CONSTANT .GT. 10 CHARACTERS, EXCESS CHARACTERS
25	I	43 CD 25	FIELD WIDTH OF A CONVERSION DESCRIPTOR SHOULD BE AS LARG

SYMBOLIC REFERENCE MAP (R=3)

ENTRY POINTS		DEF LINE	REFERENCES
4134	TABLE	1	

```
FTN 4.6+452          07/14/78  09.46.16        PAGE      1

R,BETA

COUSTIC

.2,

9,10HBETA (DEG),10,0,0)

ESS CHARACTERS INITIALIZED INTO SUCCEEDING WORDS.
ULD BE AS LARGE AS THE MINIMUM SPECIFIED FOR THAT DESCRIPTOR
```

```
      SUBROUTINE TILT(PHI,GAMMA,DVOVAV,EPS,ITMAX,IER,BETA)
C
C     THIS ROUTINE COMPUTES THE PRISM ANGLE REQUIRED
C     FOR A METAL PRISM TO TILT A SURFACE WAVE BY A GIVEN ANGLE.
C
C        PHI = DESIRED TILT ANGLE (RADIANS).
C      GAMMA = POWER FLOW CONSTANT.
C        EPS = DESIRED ACCURACY.
C      ITMAX = MAXIMUM NUMBER OF ITERATIONS.
C       BETA = PRISM (BASE) ANGLE (RADIANS).
C     DVOVAV = (V0-VM)/V0, WHERE V0 = WAVE VELOCITY ON XTAL.
C                                VM = WAVE VELOCITY ON METAL PRISM.
C        IER = RETURNED ERROR INDICATOR, IER = 0 FOR NO ERROR,
C                                            = 0 NO CONVERGENCE.
C
      REAL K1,K2
      IF(PHI.EQ.0.)GO TO 110
      THETA2=PHI/(1.+GAMMA)
      SINT2=SIN(THETA2)
      COST2=COS(THETA2)
      GAMMA2=GAMMA/2.
      V2=1.+GAMMA2*THETA2*THETA2
      K2=1.-DVOVAV
      K1=2./K2
      THETA1=DVOVAV*SINT2/((K1-1.)*V2-COST2)

      DO 100 IT=1,ITMAX
      V1=1.+GAMMA2*THETA1*THETA1
      COST1=COS(THETA1)
      SINT1=SIN(THETA1)
      CSCT1=1./SINT1
      COTT1=COST1*CSCT1
      GT1=GAMMA*THETA1
      F=(K1*COST1-V1)*V2-(COST2+(COTT1-K2*V1*CSCT1)*SINT2)*V1
      FPRIME=-(K1*SINT1+GT1)*V2-(COST2+(COTT1-K2*V1*CSCT1)*SINT2)*GT1
     +(CSCT1-K2*(V1*COTT1+GT1))*CSCT1*SINT2*V1
      TNEW=THETA1-F/FPRIME
      IF(ABS((THETA1-TNEW)/THETA1).LE.EPS)GO TO 105
  100 THETA1=TNEW
      IER=1
      GO TO 107
  105 IER=0
      THETA1=TNEW
  107 CSCT1=1./SIN(THETA1)
      COTT1=COS(THETA1)*CSCT1
      V1=1.+GAMMA2*THETA1*THETA1
      BETA=ATAN(COTT1-K2*V1*CSCT1)
      RETURN
  110 BETA=3.141592654/2.
      RETURN
      END
```

```
1                                  FTN 4.6+452          07/14/78  09.46.16          PAGE      1

I,GAMMA,DVOVAV,EPS,ITMAX,IER,BETA)

TES THE PRISM ANGLE REQUIRED
TO TILT A SURFACE WAVE BY A GIVEN ANGLE.

ILT ANGLE (RADIANS).
N CONSTANT.
CCURACY.
UMBER OF ITERATIONS.
SE) ANGLE (RADIANS).
0, WHERE V0 = WAVE VELOCITY ON XTAL.
         VM = WAVE VELOCITY ON METAL PRISM.
ERROR INDICATOR, IER = 0 FOR NO ERROR,
                     = 0 NO CONVERGENCE.

 110
MA)

2*THETA2

2/((K1-1.)*V2-COST2)

1*THETA1

-(COST2+(COTT1-K2*V1*CSCT1)*SINT2)*V1
GT1)*V2-(COST2+(COTT1-K2*V1*CSCT1)*SINT2)*GT1
(V1*COTT1+GT1))*CSCT1*SINT2*V1
ME
N)/THETA1).LE.EPS)GO TO 105

1)
CSCT1
1*THETA1
*V1*CSCT1)

.
```

```
      PROGRAM BENT(INPUT,OUTPUT)
C
C
C       PROGRAM BENT CALCULATES THE EFFECT OF
C     A TILTED ACOUSTIC BEAM AT RESONANCE FOR A
C     N-PERIOD SAW TRANSDUCER
C     LOCATED A DISTANCE Z FROM THE GENERATING
C     TRANSDUCER ALL AS A FUNCTION OF ANGLE
C
C
C
C
C  INPUT VARIABLES -
C
C          Y0 = DIFFERENCE IN HEIGHTS OF THE XDUCER CENTERS.
C      OVALAP = XDUCER APERTURE, (METERS).
C       NELEM = NUMBER OF FINGERS IN SECOND XDUCER
C      DTHETA = ANGLE INCREMENT (DEGREES).
C      NTHETA = NUMBER OF ANGLES.
C     LAMBDA0 = WAVELENGTH AT RESONANCE.
C
      REAL L,LAMBDA,LAMBDA0,LOSS(500),ANGLE(500),PLTID(3),THETER(500)
      DATA PLTID/30HSZABO   X3683  PROJECT TILT
      DATA PI/3.141592654/
      NAMELIST /DATER/Y0,OVALAP,DTHETA,NTHETA,LAMBDA0,NELEM
C
      CALL PLTID3(PLTID,100.,12.,1.0)
C
C     SET UP DEFAULTS AND READ IN $DATER NAMELIST.
C
      Y0=0.
      OVALAP=500.E-6
      NELEM=605
      LAMBDA0=8.987E-6
      DTHETA=.1
      NTHETA=151

      READ DATER
      PRINT DATER
C
      LAMBDA=LAMBDA0
      L=OVALAP/LAMBDA0
      W1=LAMBDA0/4.
      TWOW1=2.*W1
      TWOPI=2.*PI
      R2=TWOW1*(NELEM-1)

      DO 400 IY0=1,11
C     Y0=(IY0-1)*1.E-4
      Y0=(IY0-1)*5.E-5
C
      PRINT 3
      PRINT 4,Y0
```

```
T,CUTPUT)

LCULATES THE EFFECT OF
 BEAM AT RESONANCE FOR A
SDUCER
E Z FROM THE GENERATING
 A FUNCTION OF ANGLE




CE IN HEIGHTS OF THE XDUCER CENTERS.
PERTURE, (METERS).
F FINGERS IN SECOND XDUCER
CREMENT (DEGREES).
F ANGLES.
TH AT RESONANCE.

BDA0,LOSS(500),ANGLE(500),PLTID(3),THETER(500)
BO  X3683  PROJECT TILT
54/
0,OVALAP,DTHETA,NTHETA,LAMBDA0,NELEM

,100.,12.,1.0)

ND READ IN $DATER NAMELIST.
```

```
          PRINT 2
      C
          Y0=Y0/LAMBDA0
      C   TILT ANGLE (THETA) LOOP BEGINS HERE.
```



```
      C
          DO 300 ITHETA=1,NTHETA
          THETAD=(ITHETA-1)*DTHETA
          THETA=THETAD*.01745329252
          SINTH=SIN(THETA)
          COSTH=COS(THETA)
          PHI=TWOPI*OVALAP/LAMBDA*SINTH
          BIGGIE=L/COSTH

          SIGN=1.
          A1=(Y0-R2/2.*SINTH-L/2.)/COSTH+L/2.
          C2=SINTH/2.
          SUMR=SUMI=0.
          IF(ITHETA.EQ.1)GO TO 250

          DO 200 I=1,NELEM
          AI=A1+(I-1)*C2
          IF(ABS(AI).GT.BIGGIE)GO TO 200
          SIGMA=1.-AI/L
          ARG=SIGMA*PHI
          SUMR=SUMR+SIGN*SIN(ARG)
          SUMI=SUMI+SIGN*(COS(ARG)-1.)
      C   SIGN=-SIGN
      200 CONTINUE
          POWER=COSTH/PHI**2*(SUMR*SUMR+SUMI*SUMI)
          POWER=-20.*ALOG10(POWER)-W
          PHASE=ATAN(SUMI/SUMR)
          GO TO 289
      250 POWER=PHASE=W=0.
          SIGN=1.
          DO 278 I=1,NELEM
          AI=A1+(I-1)*C2
          IF(ABS(AI).GT.BIGGIE)GO TO 278
          SIGMA=1.-AI/L
          W=W+SIGN*SIGMA
      C   SIGN=-SIGN
      278 CONTINUE
          W=-20.*ALOG10(W*W)
      C
      C   FILL PLOT BUFFERS AND PRINT OUT RESULTS.
      C
      289 LOSS(ITHETA)=POWER
          ANGLE(ITHETA)=PHASE
          THETER(ITHETA)=THETAD
          PHASED=PHASE/.01745329252
      300 PRINT 1,THETAD,POWER,PHASED
          PRINT 2
      C
      C   PLOT POWER LOSS AND PHASE SHIFT VS. TILT ANGLE.
      C
          CALL DQPLOT(PLTID,THETER,LOSS,NTHETA,10HTILT ANGLE,10,9HLOSS (DB),
         $   9,0,0)
      C   CALL COPLOT(PLTID,THETER,ANGLE,NTHETA,10HTILT ANGLE,10,
```

```
      PRINT 2

      Y0=Y0/LAMBDA0
      TILT ANGLE (THETA) LOOP BEGINS HERE.
```



```
      DO 300 ITHETA=1,NTHETA
      THETAD=(ITHETA-1)*DTHETA
      THETA=THETAD*.01745329252
      SINTH=SIN(THETA)
      COSTH=COS(THETA)
      PHI=TWOPI*OVALAP/LAMBDA*SINTH
      BIGGIE=L/COSTH

      SIGN=1.
      A1=(Y0-R2/2.*SINTH-L/2.)/COSTH+L/2.
      C2=SINTH/2.
      SUMR=SUMI=0.
      IF(ITHETA.EQ.1)GO TO 250

      DO 200 I=1,NELEM
      AI=A1+(I-1)*C2
      IF(ABS(AI).GT.BIGGIE)GO TO 200
      SIGMA=1.-AI/L
      ARG=SIGMA*PHI
      SUMR=SUMR+SIGN*SIN(ARG)
      SUMI=SUMI+SIGN*(COS(ARG)-1.)
      SIGN=-SIGN
200   CONTINUE
      POWER=COSTH/PHI**2*(SUMR*SUMR+SUMI*SUMI)
      POWER=-20.*ALOG10(POWER)-W
      PHASE=ATAN(SUMI/SUMR)
      GO TO 289
250   POWER=PHASE=W=0.
      SIGN=1.
      DO 278 I=1,NELEM
      AI=A1+(I-1)*C2
      IF(ABS(AI).GT.BIGGIE)GO TO 278
      SIGMA=1.-AI/L
      W=W+SIGN*SIGMA
      SIGN=-SIGN
278   CONTINUE
      W=-20.*ALOG10(W*W)

      FILL PLOT BUFFERS AND PRINT OUT RESULTS.

289   LOSS(ITHETA)=POWER
      ANGLE(ITHETA)=PHASE
      THETER(ITHETA)=THETAD
      PHASED=PHASE/.01745329252
300   PRINT 1,THETAD,POWER,PHASED
      PRINT 2

      PLOT POWER LOSS AND PHASE SHIFT VS. TILT ANGLE.

      CALL DQPLOT(PLTID,THETER,LOSS,NTHETA,10HTILT ANGLE,10,9HLOSS (DB),
     $   9,0,0)
      CALL COPLOT(PLTID,THETER,ANGLE,NTHETA,10HTILT ANGLE,10,
```

```
C          11HPHASE ANGLE,11,0,0)
  400 CONTINUE
      CALL ENDPLT
```

```
      PROGRAM BENT          74/74   OPT=1

115             1 FORMAT(3F10.2)
                2 FORMAT(1X20(*-*))
                3 FORMAT(*1*4X*THETA*6X*LOSS*5X*PHASE*)
                4 FORMAT(20X,E15.6)
                  END
```

```
CARD NR. SEVERITY  DETAILS    DIAGNOSIS OF PROBLEM

   23       I      SZABO   X  HOLLERITH CONSTANT .GT. 10 CHARACTERS,
                      EXCESS CHARACTERS INITIALIZED INTO SUCCEEDING WORDS.
```

```
      SYMBOLIC REFERENCE MAP (R=3)

 ENTRY POINTS      DEF LINE      REFERENCES
  4134  BENT            1
```

MISSION
of
Rome Air Development Center

RADC plans and executes research, development, test and selected acquisition programs in support of Command, Control Communications and Intelligence (C^3I) activities. Technical and engineering support within areas of technical competence is provided to ESD Program Offices (POs) and other ESD elements. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.